Obihai Technology, Inc.

Obihai IP Phone Administration Guide

Models:

OBi1062

OBi1032

OBi1022

OBi2062

OBi2162

OBi2182

October 2017

Audience

Internet Telephony Service Providers (ITSP), Managed Service Value Added Resellers (VAR), Internet Telephony Professionals, and Technology Hobbyists.

Note for Australian readers: Throughout this document we refer to ITSP – treat this term the same as you would for VSP (Voice Service Provider).

Note to End Users

End users are highly encouraged to use the OBiTALK web portal to configure and manage their OBi devices.

Visit www.obitalk.com to create an OBiTALK account and within the portal. Once logged in, click "Add Device" to add your OBi Phone to the portal. You can change all the advanced settings within this guide via the portal by using "OBi Expert" mode. To enter OBi Expert, click on your device and at the next page you'll see a link to enter the OBi Expert Configuration page. Within this page you can then click another link to bring up the device's complete set of configuration parameters.

You can enable 1-click expert access from the OBi Dashboard by enabling this option under your user settings within the portal (This option is off by default).

Where to Go for Help

Obihai has a number of options available to customers who are seeking help regarding their Obihai products.

- Visit the OBiTALK Documents and Guides found at www.obihai.com/docs-downloads
    - OBi Phone Administration Guide
    - OBi Device Provisioning Guide
    - OBiTALK Device Management Platform: Frequently Asked Questions
- E-mail the Obihai Service Provider Support Team at: spsupport@obihai.com
- Call us at the Obihai Service Provider Support line: +1 (408) 890-6000

Notational Conventions

A device configuration parameter and its value is represented in the format:

***Parameter Group Name*::ParameterName** = `Parameter Value`

***Parameter Group Name*::ParameterName** = {*replace-this-with-actual-value*}

Parameter Group Name is the heading of the parameter group on the left side panel of the device local configuration or "OBiTALK Configuration" web page and may contain spaces. When a group heading has more than one level, each level is separated with a – , such as:

***Services Providers - ITSP Profile A – SIP*::**

ParameterName is the name of the parameter as shown on the web page and MUST NOT CONTAIN ANY SPACES. `Parameter Value` is the literal value to assign to the named parameter and may contain spaces. ***Group Name*** or its top-level headings may be omitted when the context is clear. For example:

> ***SP1 Service*::AuthUserName** = `4082224312`
>
> ***ITSP Profile A - SIP*::ProxyServer** = `sip.myserviceprovider.com`
>
> **ProxyServerPort** = `5082`

[*optional values*]

Boolean Values

Parameters that take a Boolean (true or false) value can be identified on the phone native configuration web pages (or OBi Expert) by a check box / tick box (instead of an input-box or drop-down list) next to the parameter name. Throughout the document we may loosely refer to a Boolean value as enable/disable or yes/no, but the only valid Boolean parameter values to use in a phone configuration file that is recognized by the phone is either `true`/`false` or `True`/`False` (case-sensitive). This is equivalent to checked/unchecked on the configuration web pages.

Multiple Choice Values

Parameters that take one of several valid options from a drop-down list on the device message must be provisioned with string values that match exactly one of those choices. Otherwise the device will use the default choice. The matching of provisioned value against valid strings is case-sensitive and does not allow extra spaces.

Parameter Values

When entering a parameter value from the web page or via provisioning, you should avoid adding extra white spaces before or after the parameter value. If the value is a comma separated list of strings, or may contain attributes after a comma or semi-colon, etc, you should also avoid adding extra white space before and after the delimiter. For example:

CertainParameter = `1,2,3,4;a;b;c`

If a parameter value can include white spaces, such as **X_DisplayLabel**, you should use just a single space character and no extra space before and after the value. For example:

X_DisplayLabel = `My New Service`

Introduction

The Obihai family of IP Phones, including the OBi1022, OBi1032, OBi1062, OBi2062, OBi2162, and OBi2182 support HD Voice with a full-duplex speakerphone, a high-resolution color active-matrix TFT LCD display with a customizable user interface, as well as a large number of fully programmable Feature Keys.

All OBi IP Phone models share the same functionalities:

- Support all standard SIP-based IP PABX and ITSPs/VSPs
- Suited for all service provider and enterprise deployment environments, regardless of size
- For do-it-yourself (DIY) or self-service installations, anyone regardless of whether they are a home user, small business, or a corporate IT department can easily install, setup and manage the OBi IP Phone
- Integrate seamlessly with popular softswitch architectures such as BroadSoft, FreePBX, Asterisk, Elastix, Metaswitch and FreeSwitch, to name a few
- Cloud management via OBiTALK.com with both a user portal as well as an ITSP partner portal (what we call the Device Management Platform (DMP) with an optional RESTful API

OBi IP Phone Hardware

This section summarizes the hardware characteristics of each OBi IP Phone. The following table provides a specification summary for each model, while figures 1 through 4 show the front and back of each OBi IP Phone as well as the phone with an attached OBi1000e and OBi2000e Sidecar.

	OBi1062	OBi1032	OBi1022	OBi2062	OBi2162	OBi2182
LCD Display	480x272, 4.3” Color TFT	480x272, 4.3” Color TFT	320x160, 3.5” Color TFT	800x480, 5” Color TFT	800x480, 5” Color TFT	800x480, 5” Color TFT
Ethernet Ports	2x GigE	2x Fast Ethernet	2x Fast Ethernet	2x GigE	2x GigE	2x GigE
PoE	Yes	Yes	Yes	Yes	Yes	Yes
Line Keys/Virtual Line Keys	6/24	3/12	5/10	3/12	3/12	6/24
Left Line Keys/Virtual Line Keys	0	0	0	3/12	3/12	6/24
Programmable Keys	8	8	1	0	0	0
Programmable Hard Keys	0	0	0	9	9	9
USB 2.0 Ports	2x USB 2.0	2x USB 2.0	1x USB 2.0	2x USB 2.0	2x USB 2.0	2x USB 2.0
3.5mm Headset Jack	Yes	Yes	Yes	Yes	Yes	Yes
RJ9 Headset Port	Yes	Yes	No	Yes	Yes	Yes
Electronic Hook Switch (EHS) Support	Yes (requires optional OBiEHS)	Yes (requires optional OBiEHS Kit)	No	Yes (requires optional OBiEHS)	Yes (requires optional OBiEHS)	Yes (requires optional OBiEHS)
WiFi	Yes (built-in for 2.4G; requires OBiWiFi USB Adapter for Dual Band)	Yes (requires OBiWiFi USB Adapter)	Yes (requires OBiWiFi USB Adapter)	Yes (requires OBiWiFi USB Adapter)	Yes (built-in Dual Band WiFi)	Yes (built-in Dual Band WiFi)
Bluetooth	Yes (built-in)	Yes (requires OBiBT USB Adapter)	Yes (requires OBiBT USB Adapter)	Yes (built-in; supports HD Voice)	Yes (built-in; supports HD Voice)	Yes (built-in; supports HD Voice)

FXO Telco Line Service	Yes (requires OBiLINE USB Adapter)	Yes (requires OBiLINE USB Adapter)	No	Yes (requires OBiLINE USB Adapter)	Yes (requires OBiLINE USB Adapter)	Yes (requires OBiLINE USB Adapter)
HD Voice	Yes	Yes	Yes	Yes	Yes	Yes
HD Full-Duplex Speakerphone	Yes	Yes	Yes	Yes	Yes	Yes
Message Waiting Light	Yes	Yes	Yes (via Programm-able Key)	Yes	Yes	Yes
Sidecar* Support * Sidecars are available separately from Obihai. Each sidecar provides 16 additional feature keys – Order Obihai model OBi1000e	Yes, 16 Keys per sidecar (up to 2)	Yes, 16 Keys per sidecar (up to 2)	No	No	No	No

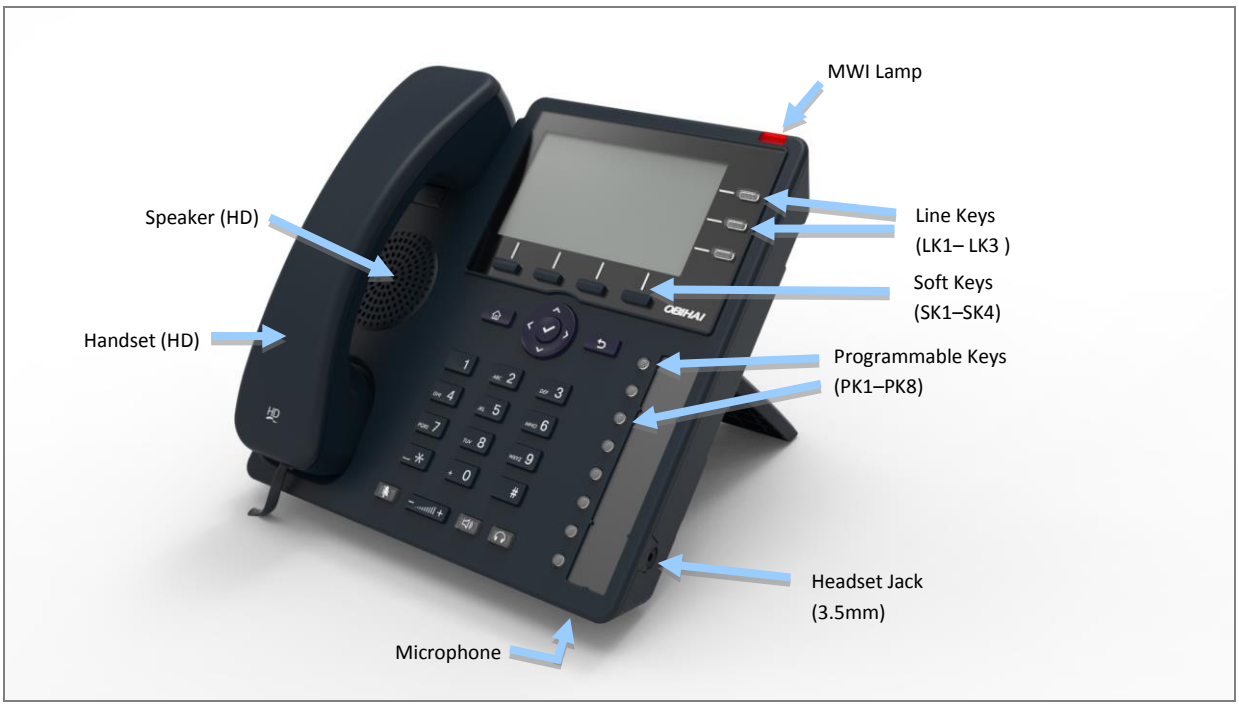


Figure 1: Front of the OBi1032 IP Phone

Figure 2: Front of the OBi1062 IP Phone

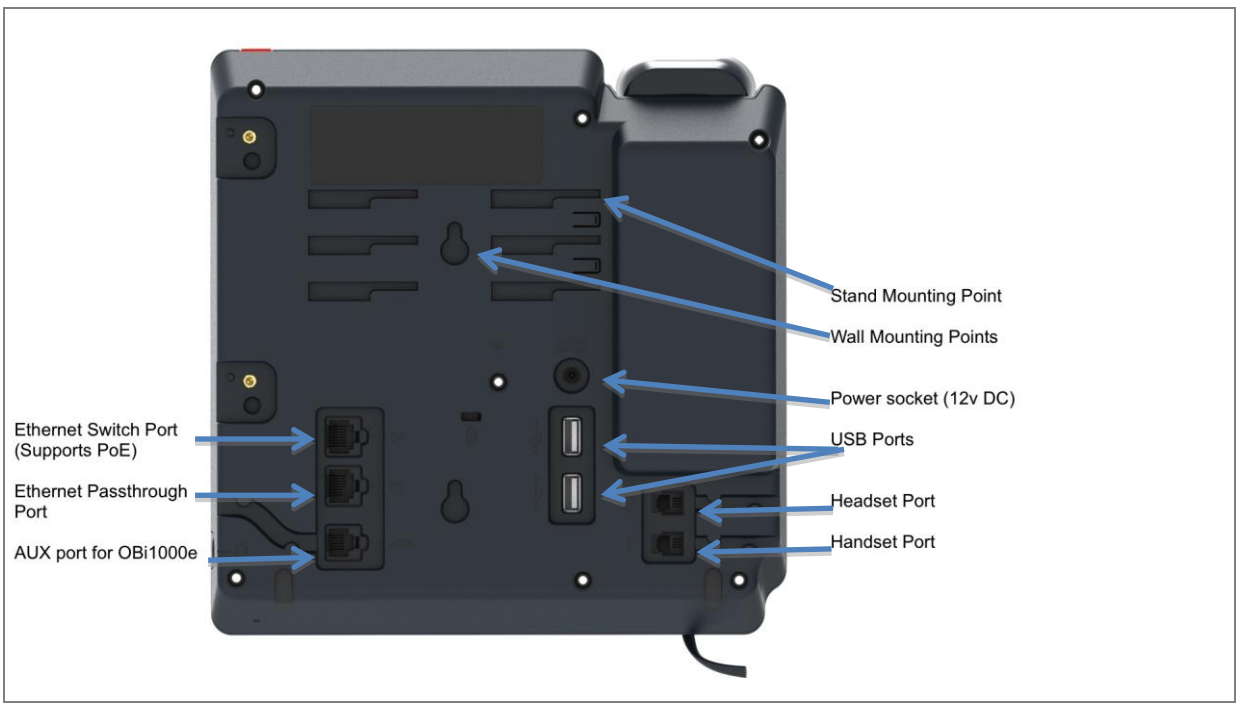


Figure 3: Rear of the OBi1032 and OBi1062 IP Phones

Figure 4: OBi1062 with an attached OBi1000e Sidecar (Option Not Available on OBI1022 and OBi2000 Series)

Figure 5: Front of the OBi1022 IP Phone

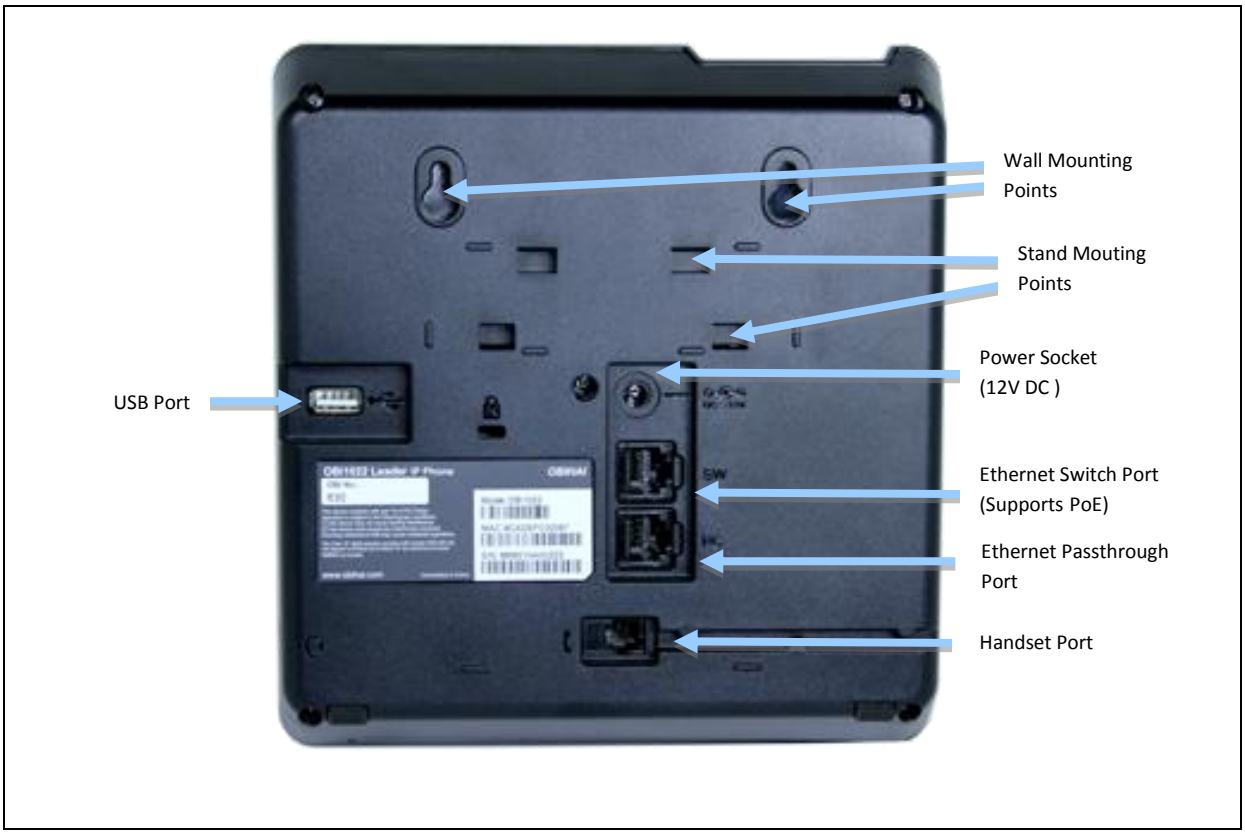


Figure 6: Rear of the OBi1022 IP Phone

Figure 7: Front of the OBi2182 IP Phone

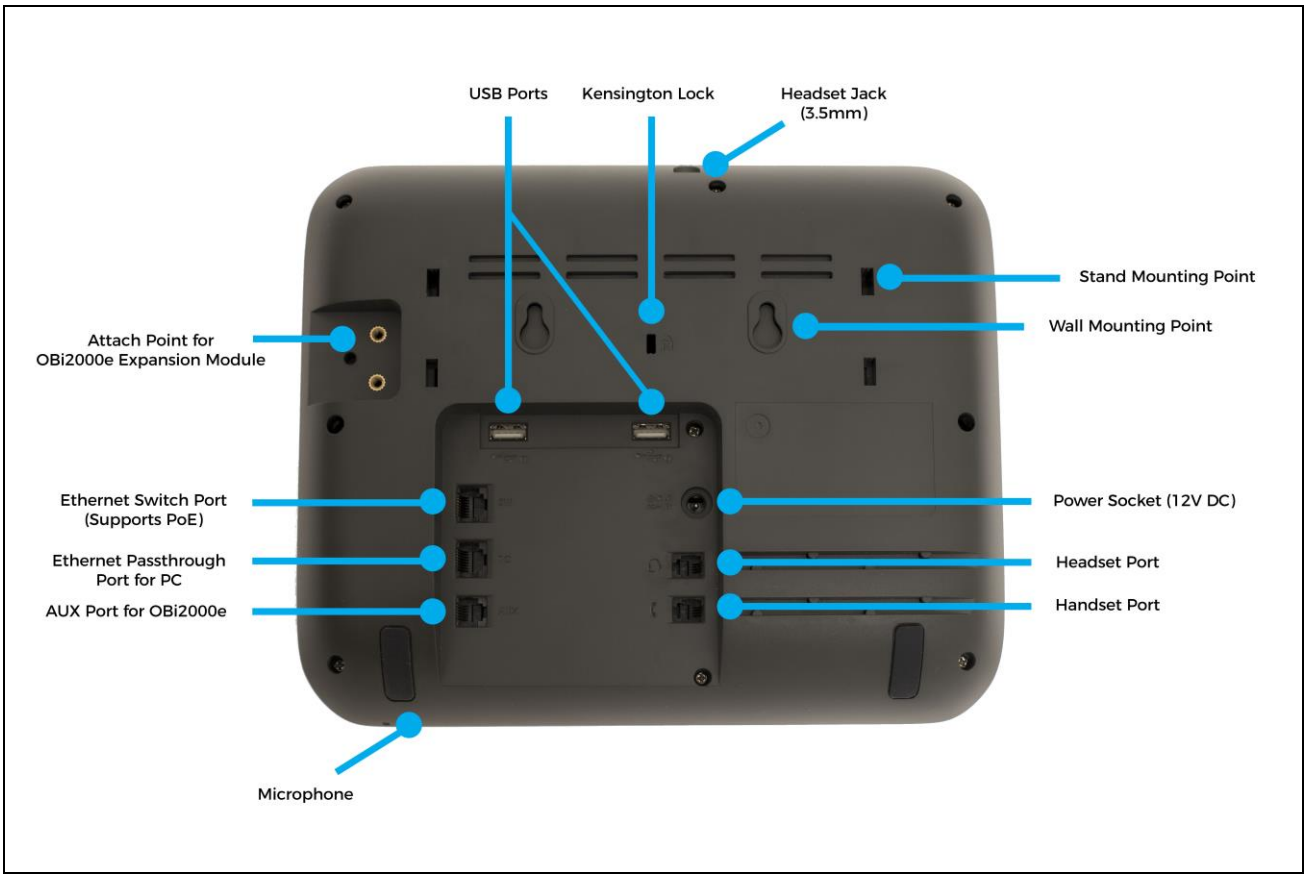


Figure 7: Back of the OBi2182 IP Phone

Accessories Available Separately from Obihai

The OBi IP Phone supports the following accessories:

- OBiWiFi5G USB Adapter: To connect the OBi IP Phone to a WiFi network
- OBiBT USB Adapter: To attach Bluetooth devices such as headsets and mobile phones
- OBiLINE FXO Adapter (Not available on OBi1022): To connect to a Plain Old Telephone System (POTS), for example, to connect the OBi to the PSTN via phone jack provided by your telephone company or to connect to an analog PABX
- OBi1000e Sidecar (For OBi1032 and OBi1062 only): To provide 16 additional feature keys. Up to two sidecars may be daisy chained from the auxiliary (AUX) port on the rear of the OBi1032 or OBi1062.
- OBi2000e Sidecar (For OBi2000 Series models only): To provide 24 additional line keys with labels displayed by an LCD screen. Up to two sidecars may be daisy chained from the auxiliary (AUX) port at the attach points on rear of the OBi2000 Series IP Phone.
- USB Headsets
- USB Keyboard: For text entry using a standard keyboard

Other Accessories

The OBi IP Phone also supports the following storage devices for copying ring tones, background images and other media to the device:

- USB Disk Drives
- USB Thumb Drives

Connecting the OBi IP Phone

The OBi can be powered in two ways:

- Using an Ethernet switch that supports Power over Ethernet (PoE): Connect the Ethernet Switch Port (marked SW) on the rear of the OBi to a PoE Ethernet switch using a Category-5 (or better) Ethernet cable
- Connecting an OBiPA 12V DC Power Adapter: Plug the 12V jack into the 12V socket on the back of the OBi IP Phone, then connect AC (mains) adapter to the AC (mains) power socket.

Connecting the Phone to the Network

You must connect the phone to a wired LAN or WiFi network in order to obtain phone service. In most cases your network will require an internet connection and your service provider will set out requirements for the required capacity of your connection to support voice services. In some cases the OBi IP Phone may be deployed on a LAN or WAN with no Internet access – for example in a corporate environment within a voice-only VLAN.

Connecting to the LAN Over Wired Ethernet

Connect a Category-5 (or better) Ethernet cable from an available switch port to the RJ45 Port labeled SW on the back of the phone. Note that the SW port also supports PoE - if it is connected to a standard PoE switch port, it can draw power from the switch without needing to connect to an OBiPA 12V DC power adapter (sold separately).

Connecting to the WLAN Over WiFi

The OBi IP Phone can also connect to a phone service over a WiFi network. The OBi2162 and OBi2182 phones have built-in dual-band WiFi (802.11ac 2.4/5GHz), the OBi1062 has built-in WiFi (802.11n 2.4GHz), whereas the OBi1022, OBi1032 and OBi2062 can connect to WiFi using the OBiWiFi5G accessory. The phone must join the wireless network by connecting to a WiFi Access Point (AP). The user may use the WiFi setup utility within the Settings app to scan for APs that are nearby, identify the correct AP by its broadcast SSID (WiFi network name) and connect to it. If security is enabled on the AP, an input prompt will pop up on the screen to let you enter the WiFi access password. Upon entry of correct credentials and connection to WiFi the network, the screen will display an icon on the status bar that indicates the access point is connected and also shows the signal strength. For further setup information, read “WiFi Setup” in the “OBi Phone Apps” section of this document.

Overview of Phone Features

Administrative Features

- Web pages of phone status and configuration of all parameters
- Remote Provisioning
- OBiTALK Provisioning
- Automated Firmware Update

Voice Features

- Six (6) SIP or Google Voice (GV) Accounts
- Universal inter- and intra service two-way call bridging among the 6 SIP/GV services, the OBiTALK service, and the OBiBluetooth service
- Universal intra- and inter-service call transfer and call forward by local call bridging
- Automatic Attendant with customizable audio prompts that may be recorded directly on the phone
- SIP Support for Voice Over IP
- OBiTALK Managed VoIP Network for OBi Endpoint Devices & Applications
- High Quality Voice Encoding Using G.711, G.726, G.729, G.722, iLBC, and OPUS Algorithms
- Recursive Digit Maps and Associated Call Routing (Outbound and Inbound)

Call Features

- Message Waiting Indication - Visual and Tone Based
- Four Way Conference Calling with Local Mixing
- Caller ID and Calling Line ID Presentation
- Call Waiting
- Call Forward - Unconditional
- Call Forward on Busy
- Call Forward on No Answer
- Call Transfer
- Call Park
- Anonymous Call
- Block Anonymous Call
- Do Not Disturb
- Call Return
- Repeat Dialing
- Multicast Paging Groups
- Music On Hold

Soft Switch Support

- BroadSoft
- MetaSwitch
- FreeSwitch
- SkySwitch
- Asterisk

Integrated GUI Applications

- Phone Book
- Call History
- 3rd Party XML Apps

Complementary Obihai Products and Services

OBi1000 IP Phones are complemented by other OBi Products & Services:

OBiTALK: A customer portal for device management allowing members to add people and associated OBi endpoints to "circles of trust" such that additional functionality can be shared amongst authorized users.

OBiTALK Device Management Platform (DMP): A cloud-based management tether to the OBi endpoints for secure and remote provisioning, real-time call status reporting and troubleshooting, device UI management and more. Taking in a form of a web portal and an API, the OBiTALK DMP provides access to the configurations of the multiple endpoints regardless of location, making it as if they are virtually sitting right at your desk.

The OBiTALK DMP is designed for use of service providers, system integrators and value-added resellers (VAR) deploying OBi Universal Adapters and IP Phones. When a user is logged in, they can manage all their OBi endpoints remotely, enabling fast and easy device deployment and maintenance.

Contact partner@obihai.com to learn more about the OBiTALK DMP, and how to sign up to receive access.

Configuration and Management Interfaces

There are several ways to configure and manage the OBi IP Phone. Use the method (or combination of methods) that best suit your deployment scenario.

Device Local Configuration

The OBi IP Phone has an integrated device management web server that can be accessed from any standard (desktop or tablet) web browser. Although all popular browsers are tested for compatibility with the OBi device management web server, there may be inconsistencies that arise from time to time. Please contact spsupport@obihai.com if you have any questions about the OBi device management web server and how it appears in your browser window.

To access the OBi IP Phone Device Management Web Page:

1. Connect the phone to the LAN
2. From the phone Main Menu, select Settings
3. Under Settings, the first item, Network, shows the IP address of the phone
4. Enter the phone IP Address as the URL of the web site you want to visit in your web browser
5. When prompted by the web browser, enter `admin` for user name and `admin` for password. Note that the password is the standard factory default value. If the value has been changed, enter the correct value instead.

When you access the OBi device management web page, you may be prompted to enter a user name and password. There are two levels of access to the OBi web page – User Level and Admin Level. The default "user name / password" for the User Level access is "user / user". The default "user name / password" for the Admin Level access is "admin / admin". The Admin and/or User passwords may have been changed using the OBi device web page, provisioning by a service provider or via the OBiTALK web portal (Admin only). Please be sure you have access to the correct Admin or User password before you attempt to log on to the OBi Device Management Web Page.

The OBi device management web page is organized into sections to allow for a manageable and compartmentalized approach to configuring the many hundreds of parameters available on the OBi device. Use the expandable / collapsible menu tree on the left side of the page to easily navigate the various configuration parameter sections of the OBi device.

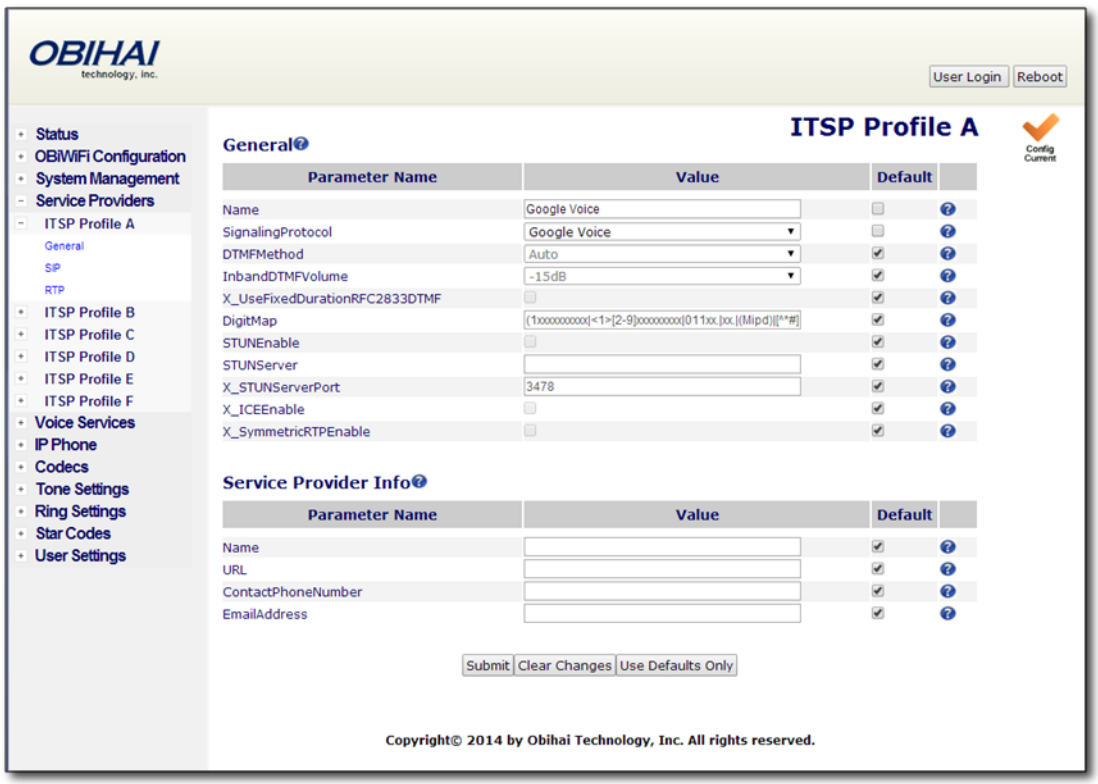
IMPORTANT: Every configuration page must be submitted individually after changes made on the page. Otherwise those changes will be discarded once you navigate to another page. Most changes will require a reboot of the unit (by clicking the reboot button for instance) to take effect. However, you may reboot the unit just once after you have made and submitted all the necessary changes on all the pages.

Web Page Conventions and Icons & Buttons:

Icon / Button	Description	Remark
	This icon indicates that there is more information available which might describe the workings, limits or thresholds for the parameter to which it is adjacent. You can mouse over this icon to reveal this information.	
Submit	When a modification has been made to a parameter on a page, the Submit button MUST be clicked before proceeding to another page.	
Clear Changes	If you make changes to a parameter on a page and you do not want to keep them for submission, click the "Clear Changes" button to revert back to the parameter setting present before the most recent change was entered.	
Use Defaults Only	Click the "Use Defaults Only" button if you want to revert all parameters on a given page to their Default settings. If you want to revert just one or two parameters on a page to default settings you should use the Default check box found on the right side of the parameter. See next Item.	You will be prompted to confirm that you want all the parameters on the page to revert back to system default settings.
Default	When you wish to modify a parameter away from its default setting, you should un-check the 'Default' box. This will open the parameter field for access and modification. If there is a non-default setting in a parameter field and you want to revert that parameter back to its default setting, check the "Default" box and the default setting will appear.	Default value of a parameter may be changed with a firmware upgrade. Leaving a parameter at default setting allows the device to use proper default value with the firmware currently installed in the device
Config Current	This icon indicates that the configuration currently programmed on the OBi device is "set" and "running". No reboot is necessary if you have submitted configuration modifications.	This icon does not indicate the currently running configuration is working properly.
Reboot Required	After Submitting changes to a web page on the OBi, the "Reboot Required" icon may appear. In order for the modifications to run, you will need to reboot the OBi.	You can continue to make modifications to OBi parameters – on separate pages if necessary – before you reboot and "set" the modifications in the running system.
Reboot	The "Reboot" button is used when the "Reboot Required" icon appears indicating the OBi device requires a reboot to invoke one or more parameter modifications.	When performing a System Configuration Reset, the Reboot button does not need to be pressed. The OBi will reboot automatically when the "Reset" button is selected.

Local Configuration Web Page Layout

There are many configurable parameters available on the OBi1000. These parameters are organized into a number of device configuration web pages. By browsing through the web pages you can discover all the parameters that can be configured, then read or set their values. Each web page is divided into three frames: A top frame with the Obihai banner (which can be customized by the administrator), a left frame that lists the links to the available pages, and a main frame that shows the parameters of the currently selected page. An example of a device configuration web page is shown below.



Below is the list of available device configuration web pages:

Status

- System Status
- Call Status
- SP Services Stats

OBiWiFi Configuration

- WiFi Settings
- WIFi Scan

System Management

- WAN Settings
- Auto Provisioning
- Device Admin
- Device Update

Service Providers

ITSP Profile A (repeated for ITSP Profile B, C, D, E, and F)

- General
- SIP
- RTP

Voice Services

- SP1 Service
- SP2 Service
- SP3 Service
- SP4 Service
- SP5 Service
- SP6 Service
- OBiTALK Service
- Auto Attendant
- Gateways and Trunk Groups
- OBiBluetooth

IP Phone

- Phone Settings
- Line Keys
- Left Line Keys *(OBi2000 Series only)*
- Programmable Keys
- Side Car 1
- Side Car 2
- Soft Keys
- LED Settings
- LDAP
- Feature Key Customization

Codec Profiles

- Codec Profile A
- Codec Profile B

Tone Settings

- Tone Profile A
- Tone Profile B

Ring Settings

- Ring Profile A
- Ring Profile B

Star Codes

- Star Code Profile A
- Star Code Profile B

User Settings

- User Preferences
- Speed Dials
- User Defined Digit Maps

Phone Screen Capture

From the phone web page, you can take a snapshot of the current phone screen and save it as a a bmp file on your host computer. Here is how:

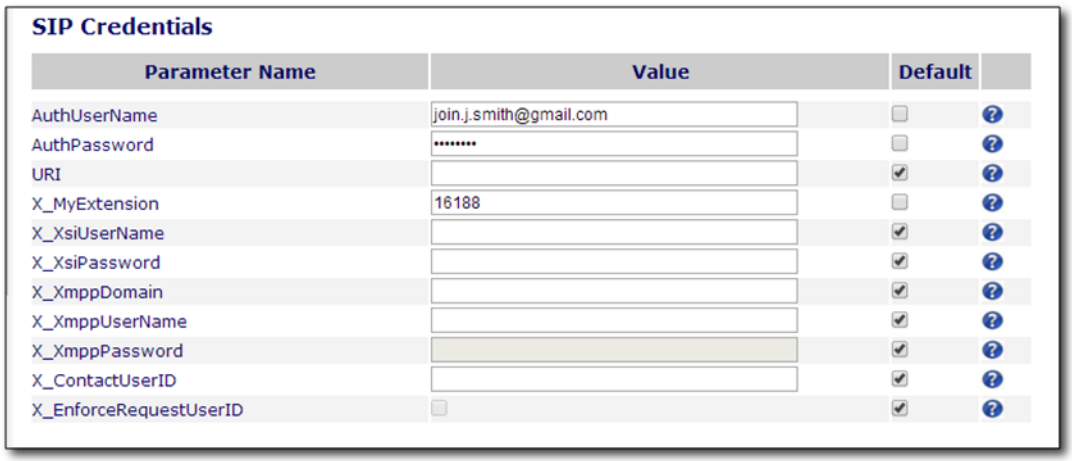
- Enable the option ***Device Admin – Web Server***::`LCDScreenShot`
- Open the **System Management – Device Update** web page of the phone and press the *Snap* button

Remote Provisioning

This is the process by which the OBi downloads a configuration file from a server, which may be located in the cloud or in the same enterprise. The configuration file may contain all the necessary parameter values for the phone to function normally, it may also tell the device to download an additional configuration file from a different URL, or to download a different firmware to replace the current one, and so on. The configuration file format and parameter naming conventions are proprietary to Obihai but are common across all Obihai products.

There are currently two configuration file formats supported: A full XML format with the XML tags in full text and a short XML format with the XML tags substituted with a single letter abbreviation. The XML structure and parameter naming convention closely follows TR-104. For a full description of the configuration file and parameter names, please refer to the *OBi Device Provisioning Guide*.

Similar to the way parameters are grouped under different device configuration web pages, parameters are grouped into a number of configuration objects for remote provisioning. In fact you will find a near one-to-one correspondence between these objects and their location within the configuration web pages. To illustrate this, consider the web page SP1 Service, the SIP Credentials section:



SIP Credentials

Parameter Name	Value	Default	
AuthUserName	join.j.smith@gmail.com	☐	?
AuthPassword	••••••••	☐	?
URI		☑	?
X_MyExtension	16188	☐	?
X_XsiUserName		☑	?
X_XsiPassword		☑	?
X_XmppDomain		☑	?
X_XmppUserName		☑	?
X_XmppPassword		☑	?
X_ContactUserID		☑	?
X_EnforceRequestUserID		☑	?

The corresponding configuration object in a phone configuration XML file is:

`VoiceService.1.VoiceProfile.1.Line.1.SIP.`

as shown below:

```
<Object>
  <Name>VoiceService.1.VoiceProfile.1.Line.1.SIP.</Name>
  <ParameterValueStruct>
    <Name>AuthUserName</Name>
    <Value>john.j.smith@gmail.com</Value>
  </ParameterValueStruct>
  <ParameterValueStruct>
    <Name>AuthPassword</Name>
```

```
      <Value>zYz123#$12</Value>
    </ParameterValueStruct>
    <ParameterValueStruct>
      <Name>URI</Name>
      <Value X_UseDefault="Yes"/>
    </ParameterValueStruct>
    <ParameterValueStruct>
      <Name>X_MyExtension</Name>
      <Value>16188</Value>
    </ParameterValueStruct>
    <ParameterValueStruct>
      <Name>X_XsiUserName</Name>
      <Value X_UseDefault="Yes"/>
    </ParameterValueStruct>
    <ParameterValueStruct>
      <Name>X_XsiPassword</Name>
      <Value X_UseDefault="Yes"/>
    </ParameterValueStruct>
    <ParameterValueStruct>
      <Name>X_XmppDomain</Name>
      <Value X_UseDefault="Yes"/>
    </ParameterValueStruct>
    <ParameterValueStruct>
      <Name>X_XmppUserName</Name>
      <Value X_UseDefault="Yes"/>
    </ParameterValueStruct>
    <ParameterValueStruct>
      <Name>X_ContactUserID</Name>
      <Value X_UseDefault="Yes"/>
    </ParameterValueStruct>
    <ParameterValueStruct>
      <Name>X_EnforceRequestUserID</Name>
      <Value X_UseDefault="Yes"/>
    </ParameterValueStruct>
  </Object>
```


Note that the dot (.) at the end of the object name is part of the name that must not be omitted in the XML file. You must use the correct object name in order to create a valid configuration file for the phone. You can find the object name corresponding to each configuration web page/section listed at the end of this document.

About ZT (Zero Touch): Device Customization at Obihai's Factory

When products are shipped from the factory, they come with a set of default parameter values installed by Obihai for all customers. A service is also available from Obihai such that products shipped to a particular customer can have a small number of parameter values customized for that customer. For example, a very useful parameter to customize is the ***ITSP Provisioning***::**ConfigURL** parameter which tells the phone where to download a configuration file. With this, the first time a new phone is powered on and connected to the network, it can automatically contact the designated URL to get the initial configuration file; hence the name "Zero-Touch".

Note: ZT devices must contact OBiTALK.com one time to get the customized values before they can start normal operation. Make sure the device can access the Internet before first use.

OBiTALK Portals

OBiTALK.com is a device management portal website to serve Obihai customers and service providers deploying OBi devices. OBiTALK.com uses remote provisioning to manage OBi devices; it stores, or dynamically generates on demand, a configuration file for each managed device which periodically checks in with the OBiTALK server for configuration updates.

There are two levels of management portals at OBiTALK.com: **User** and **ITSP**.

User Portal

Users may add one or more OBi devices to their OBiTALK account to be managed. The portal has setup wizards that help the user configure voice services on any of their devices. Users can also see the detailed status and current parameter values of their devices and easily change settings on multiple devices from within the portal. The User Portal has an upper limit of 20 devices per portal instance.

Device Management Platform (DMP) Portal – for ITSPs

Service providers and system Integrators can add devices to the OBiTALK portal and manage them in ways similar to regular users. Zero Touch (ZT) devices, on the other hand, are added to the corresponding customer's ITSP account automatically, as soon as the earmarked units are manufactured at the factory. ZT device customers can monitor the status of their units on the portal and check if (and when) new units have contacted the ZT server at OBiTALK.com and are successfully customized.

In addition, service providers and system integrators using the OBiTALK DMP portal can configure their devices on OBiTALK.com in batch mode by defining XML base profiles that may be applied to many units. When a base profile is changed, all units using this base profile will be automatically updated with any changes that have been made.

Telephone-IVR-Based Local Configuration

The OBi1000 has a built in IVR for checking and setting a small but essential subset of configuration parameters. Configuration via the IVR is a legacy configuration method inherited from older OBi products that do not have a display (such as the OBi202 and OBi508). It is included here nevertheless for additional convenience and also so that customers who are already familiar with using the OBi IVR can perform basic configuration tasks without learning new specifics about the phone first.

The IVR is, in essence, an instance of an automated attendant (AA). The OBi offers two instances of AA; referred to as AA1 (or just AA where the suffix 1 is implied) and AA2. The IVR for configuration purposes is AA2, which we will just refer to as the IVR to avoid confusion with AA1, which is the AA used to handle phone calls. AA is covered as the subject of a later section.

To invoke the IVR, the user picks the phone, dials * * * and follows the announced instructions. In order for the * * * number to work, make sure the digit map pattern `***` is included in the ***Phone Settings***::**DigitMap** parameter, and the rule, `{***:aa2} is` included in the ***Phone Settings***::**OutboundCallRoute** parameter. The standard (non-customized) default values of these parameters are, respectively:

```
([1-9]x?*(Mpli)|[1-9]S9|[1-9][0-9]S9|***|**0|**8(Mbt)|
**1(Msp1)|**2(Msp2)|**3(Msp3)|**4(Msp4)|**9(Mpp)|(Mpli))
```

and

```
{([1-9]x?*(Mpli)):pp},{**0:aa},{***:aa2},{(<**1:>(Msp1)):sp1},
{(<**2:>(Msp2)):sp2},{(<**3:>(Msp3)):sp3},{(<**4:>(Msp4)):sp4},
{(<**8:>(Mbt)):bt},{(<**9:>(Mpp)):pp},{(Mpli):pli}
```

For the meaning of these values, please see the section Digit Maps and Call Routing. Some parameter changes require a reboot to take effect. Changes from the OBiTALK Configuration will trigger the phone to reboot automatically (but will wait for any current calls to end first).

Key Ahead: By pressing the appropriate button sequence on the telephone key pad, you can barge into the next menu of the IVR or invoke a command without first waiting for the previous announcement to end.

Main Menu

The Main Menu after starting the IVR is a list of operations that can be selected by entering the corresponding 1-digit option number, as listed below:

Selection	Announcement	What Can You Do?
1	Basic Network Status Your IP address and DHCP status will be read back to you.	Press 0 to repeat the information.
2	Advanced Network Status Your primary & back-up DNS server, primary & back-up NTP server will be read back to you.	Press 0 to repeat the information.
3	DHCP Current Value Your current value will be read back to you and you will be given the option to change the value	Press 1 to enter a new value. Press 2 to set the default value. Press 0 to repeat the information.
4	IP Address Current Value Your current value will be read back to you and you will be given the option to change the value. If you elect to enter a new value (static IP address) DHCP will be disabled.	Press 1 to enter a new value. Press 2 to set the default value. Press 0 to repeat the information.
5	Password Current Value Your current IVR password value will be read back to you and you will be given the option to change the value.	Press 1 to enter a new value. Press 2 to set the default value. Press 0 to repeat the information.
6	Please Wait (while OBi is checking for software update)… This is followed by either: - Software Update Available. Press 1 to update software, OR - Software Update Not Available	If an update is available, press 1 to proceed with the update. The software update process will start as soon as you hang up the phone. Warning: Once the software upgrade process starts, the device's power LED will blink rapidly. Please make sure the power and network cable stay connected to the unit until the process is complete.
8	Restore Factory Default	Press 1 to confirm device restore to factory default settings. Press # to return to device configuration menu. Press # # to exit IVR.

9	Reboot OBi Device	Press 1 to confirm device reboot. Press # to return to device configuration menu. Press # # or hang up to exit IVR.
0	Additional Options Access other configuration options of the OBi device.	Enter option followed by the # key.

Additional Options (Menu 0)

There are many additional options beyond the top level options 1-9. Unlike the top level options, however, the list of available additional options are not announced. The user must enter the corresponding option number followed by a # key to select the particular option. The available additional options are listed in the tables below (grouped by function):

System Level Options

Selection (Always Press "#" After Entering Selection)	**Announcement**	**What Can You Do?**
1	Firmware Version The current value of the firmware version will be read back.	Press 0 to repeat the information. Press # to enter another configuration selection.
2	IVR Password The current value of the IVR password will be read back.	Press 1 to enter a new value. Press 2 to set the default value. Press 0 to repeat the information. Press # to enter another configuration selection.
3	Debug Level The current value of the debug level will be read back.	Press 1 to enter a new value. Press 2 to set the default value. Press 0 to repeat the information. Press # to enter another configuration selection.
4	Syslog Server IP Address The current IP address of the syslog server will be read back.	Press 1 to enter a new value. Press 2 to set the default value. Press 0 to repeat the information. Press # to enter another configuration selection.
5	Syslog Server Port The current value of the syslog server port will be read back.	Press 1 to enter a new value. Press 2 to set the default value of 514. Press 0 to repeat the information. Press # to enter another configuration selection.

Network Related Configuration Options

Selection (Always Press "#" After Entering Selection)	**Announcement**	**What Can You Do?**
20	DHCP Configuration The current value of the DHCP configuration will be read back.	Press 1 to enter a new value. Press 2 to set the default value. Press 0 to repeat the information. Press # to enter another configuration selection.

21	IP Address The current value of the IP address will be read back.	Press 1 to enter a new value. Press 2 to set the default value. Press 0 to repeat the information. Press # to enter another configuration selection.
22	Default Gateway The current value of the default internet gateway will be read back.	Press 1 to enter a new value. Press 2 to set the default value. Press 0 to repeat the information. Press # to enter another configuration selection.
23	Subnet Mask The current value of the subnet mask will be read back.	Press 1 to enter a new value. Press 2 to set the default value. Press 0 to repeat the information. Press # to enter another configuration selection.
24	DNS Server (Primary) The current value of the primary DNS server will be read back.	Press 1 to enter a new value. Press 2 to set the default value. Press 0 to repeat the information. Press # to enter another configuration selection.
26	NTP Server (Primary) The current value of the primary NTP server will be read back.	Press 1 to enter a new value. Press 2 to set the default value. Press 0 to repeat the information. Press # to enter another configuration selection.

SP1 Configuration Options

Selection (Always Press "#" After Entering Selection)	**Announcement**	**What Can You Do?**
100	Enable Service Provider One (SP1) The current value will be read back.	Press 1 to enter a new value. Press 2 to set the default value. Press 0 to repeat the information. Press # to enter another configuration selection.
101	Registration State of SP1 The current value will be read back.	Press 0 to repeat the information. Press # to enter another configuration selection.
102	SP1 User ID The current value will be read back.	Press 1 to enter a new value. Press 2 to set the default value. Press 0 to repeat the information. Press # to enter another configuration selection.
167	SP1 Block Caller ID Enable	Press 1 to enter a new value. Press 2 to set the default value. Press 0 to repeat the information. Press # to enter another configuration selection.
168	SP1 Block Anonymous Call Enable	Press 1 to enter a new value. Press 2 to set the default value. Press 0 to repeat the information. Press # to enter another configuration selection.

172	SP1 Call Forward ALL – Enable / Disable	Press 1 to enter a new value. Press 2 to set the default value. Press 0 to repeat the information. Press # to enter another configuration selection.
173	SP1 Call Forward ALL Number	Press 1 to enter a new value. Press 2 to set the default value. Press 0 to repeat the information. Press # to enter another configuration selection.
174	SP1 Call Forward on Busy – Enable / Disable	Press 1 to enter a new value. Press 2 to set the default value. Press 0 to repeat the information. Press # to enter another configuration selection.
175	SP1 Call Forward on Busy Number	Press 1 to enter a new value. Press 2 to set the default value. Press 0 to repeat the information. Press # to enter another configuration selection.
176	SP1 Call Forward on No Answer – Enable / Disable	Press 1 to enter a new value. Press 2 to set the default value. Press 0 to repeat the information. Press # to enter another configuration selection.
177	SP1 Call Forward on No Answer Number	Press 1 to enter a new value. Press 2 to set the default value. Press 0 to repeat the information. Press # to enter another configuration selection.

SP2 Configuration Options

Selection (Always Press "#" After Entering Selection)	**Announcement**	**What Can You Do?**
200	Enable Service Provider One (SP2) The current value will be read back.	Press 1 to enter a new value. Press 2 to set the default value. Press 0 to repeat the information. Press # to enter another configuration selection.
201	Registration State of SP2 The current value will be read back.	Press 0 to repeat the information. Press # to enter another configuration selection.
202	SP2 User ID The current value will be read back.	Press 1 to enter a new value. Press 2 to set the default value. Press 0 to repeat the information. Press # to enter another configuration selection.
267	SP2 Block Caller ID Enable	Press 1 to enter a new value. Press 2 to set the default value. Press 0 to repeat the information. Press # to enter another configuration selection.

268	SP2 Block Anonymous Call Enable	Press 1 to enter a new value. Press 2 to set the default value. Press 0 to repeat the information. Press # to enter another configuration selection.
272	SP2 Call Forward ALL – Enable / Disable	Press 1 to enter a new value. Press 2 to set the default value. Press 0 to repeat the information. Press # to enter another configuration selection.
273	SP2 Call Forward ALL Number	Press 1 to enter a new value. Press 2 to set the default value. Press 0 to repeat the information. Press # to enter another configuration selection.
274	SP2 Call Forward on Busy – Enable / Disable	Press 1 to enter a new value. Press 2 to set the default value. Press 0 to repeat the information. Press # to enter another configuration selection.
275	SP2 Call Forward on Busy Number	Press 1 to enter a new value. Press 2 to set the default value. Press 0 to repeat the information. Press # to enter another configuration selection.
276	SP2 Call Forward on No Answer – Enable / Disable	Press 1 to enter a new value. Press 2 to set the default value. Press 0 to repeat the information. Press # to enter another configuration selection.
277	SP2 Call Forward on No Answer Number	Press 1 to enter a new value. Press 2 to set the default value. Press 0 to repeat the information. Press # to enter another configuration selection.

OBiTALK Configuration Options

Selection (Always Press "#" After Entering Selection)	**Announcement**	**What Can You Do?**
900	Enable OBiTALK Service The current value will be read back.	Press 1 to enter a new value. Press 2 to set the default value. Press 0 to repeat the information. Press # to enter another configuration selection.
901	Registration State of OBiTALK The current value will be read back.	Press 0 to repeat the information. Press # to enter another configuration selection.
967	OBiTALK Block Caller ID Enable	Press 1 to enter a new value. Press 2 to set the default value. Press 0 to repeat the information. Press # to enter another configuration selection.

968	OBiTALK Block Anonymous Call Enable	Press 1 to enter a new value. Press 2 to set the default value. Press 0 to repeat the information. Press # to enter another configuration selection.
972	OBiTALK Call Forward ALL – Enable / Disable	Press 1 to enter a new value. Press 2 to set the default value. Press 0 to repeat the information. Press # to enter another configuration selection.
973	OBiTALK Call Forward ALL Number	Press 1 to enter a new value. Press 2 to set the default value. Press 0 to repeat the information. Press # to enter another configuration selection.
974	OBiTALK Call Forward on Busy – Enable / Disable	Press 1 to enter a new value. Press 2 to set the default value. Press 0 to repeat the information. Press # to enter another configuration selection.
975	OBiTALK Call Forward on Busy Number	Press 1 to enter a new value. Press 2 to set the default value. Press 0 to repeat the information. Press # to enter another configuration selection.
976	OBiTALK Call Forward on No Answer – Enable / Disable	Press 1 to enter a new value. Press 2 to set the default value. Press 0 to repeat the information. Press # to enter another configuration selection.
977	OBiTALK Call Forward on No Answer Number	Press 1 to enter a new value. Press 2 to set the default value. Press 0 to repeat the information. Press # to enter another configuration selection.

Auto Attendant Configuration Options

Selection (Always Press "#" After Entering Selection)	**Announcement**	**What Can You Do?**
80	Enable / Disable Auto Attendant.	Press 1 to enter a new value. Press 2 to set the default value. Press 0 to repeat the information. Press # to enter another configuration selection.

Customized AA Prompt Recording Options

Selection (Always Press "#" After Entering Selection)	**Announcement**	**What Can You Do?**
1001	Option 1001 current value is: (the recorded prompt)	Press 1 to enter a new value. Press 2 to set the default value.

		Press 0 to repeat the information. Press # to enter another configuration selection. Note: After pressing 1 to record a new prompt, the OBi says “Enter value followed by the # key”. At that point, you can press any digit (0-9) to start recording, and then press # to end recording. Tips: Leave about 1s of gap at the end of recording to avoid unintended truncation by the OBi. After a new prompt is recorded, Obi immediately plays back the recorded audio, and then presents the following options: Press 1 to save (save the recorded prompt permanently in long term memory) Press 2 to re-enter (the last recorded prompt is discarded) Press 3 to review Press # to cancel (the last recorded prompt is discarded)
Similarly for Options 1002 - 1010		

With these options you can record up to 10 prompts that can be arranged in any combination and used as customized AA prompts. Each prompt recording is limited to 60 seconds, where the prompt duration is rounded to the nearest number of seconds. A total of 122 seconds is available to store all the recordings. The device will reboot automatically when you hangup if any of the prompts have been modified and saved. Furthermore you can enter a text description for each recorded prompt as a reminder of the contents of that prompt (under the Voice Services - Auto Attendant configuration page).

Phone GUI

A limited amount of phone configuration can be done directly from the phone GUI. The most essential are the ones pertaining to getting the phone connected the network, such as IP address settings or WiFi settings.

Settings

The Phone GUI allows the following parameters to be configured under the “Settings” option of the main menu:

- Network
  - AddressingType
  - IP Address
  - Subnet Mask
  - Default Gateway
  - DNS Server1
  - DNS Server2
  - DNS Query Order

    - DNS Query Delay
    - PPPoE AC Name
    - PPPoE Service Name
    - PPPoE User Name
    - PPPoE Password
    - VLAN Enable
    - VLAN ID
    - VLAN Priority
    - 802.1X
    - 802.1X User name
    - 802..1X Password
    - LLDP-MED Enable
    - NTP Server 1
    - NTP Server 2
    - Local Time Zone
    - Daylight Saving Time Enable
    - Daylight Saving Time Start
    - Daylight Saving Time End
    - Daylight Saving Time Diff
- WiFi
    - Enable
    - OBiWiFi Setup Mode
    - SSID
    - Status
    - Security
    - Signal Strength
    - MAC Address
- Bluetooth
    - Enable
    - Status
    - Pairing Mode
    - Discoverable
- OBiLine Settings *(not available on OBi1022)*
    - Enable
    - Status
    - Directory Number
    - AC Impedance
    - TX Gain
    - RX Gain
- Storage
    - Device
    - File System Type
    - Capacity
- Programmable Keys
- Line Keys
- Left Line Keys *(available on OBi2000 series only)*
- Side Car 1 *(not available on OBi1022 and OBi2000 series)*

- Side Car 2 *(not available on OBi1022 and OBi2000 series)*
- Voice Services
    - SP1 – SP6
        - Enable
        - AuthUserName
        - AuthPassword
        - DisplayLabel
        - DisplayNumber
        - InboundCallRoute
        - RegisterEnable
        - UserAgentPort
        - SipDebugOption
        - SipDebugExclusion
        - ServProvProfile
        - URI
        - CallerIDName
        - MWIEnable
        - VMWIEnable
        - AnonymousCallBlockEnable
        - AnonymousCallEnable
        - DoNotDisturbEnable
    - ITSP Profile A – F
        - Signaling Protocol
        - ProxyServer
        - ProxyServerPort
        - ProxyServerTransport
        - OutboundProxy
        - OutboundProxyPort
        - RegistrationPeriod
        - ProxyRequire
        - DnsSrvAutoPrefix
        - DiscoverPublicAddress

- Speed Dials 99
- Device Administration
    - Web Server Port
    - Web Admin Password
    - Web User Password
    - Syslog Server
    - ITSP Provisioning Method
    - ITSP Provisioning Interval
    - ITSP Provisioning Config URL
    - Auto Firmware Update Method
    - Auto Firmware Update Interval
    - Auto Firmware Update URL

Preferences

The following options can be configured:

- Language
- Skin
- Background Picture
- Ringtone
- Screen Saver
- Screen Saver Delay in Seconds
- Screen Save Type
- Screen Brightness
- Preferred Audio Device
- Preferred Headset Device
- Do Not Disturb
- Do Not Ring
- Call Forward
- Call Waiting
- Block Anonymous Call
- Anonymous Call
- Auto Answer Page
- Join Page Group 1
- Joint page Group 2
- Ringer Volume
- Speakerphone Volume
- Speakerphone Mic Gain
- Handset Volume
- Handset Mic Gain
- RJ9 Headset Volume
- RJ9 Headset Mic Gain
- 3.5mm Headset Gain
- 3.5mm Headset Mic Gain
- BT Headset Volume
- BT Headset Mic Gain
- Equalizer
- Acoustic Echo Cancellation

Admin Password

The Voice Services and Device Administration options in the GUI are protected by the same admin password used for accessing the phone's local configuration web pages.

Networking Features

The OBi Phone offers two physical interfaces for networking: Ethernet (described as "WAN" in the configuration) and WiFi. Both interfaces may be used at the same time, but Ethernet will take precedence in ambiguous cases.

Ethernet Ports

The OBi Phone has a 3-port switch, one of which is connected internally to the phone processor for traffic to and from the OBi, with the other 2 ports exposed to the outside world via two RJ45 connectors on the back of the devices. The two exposed ports are labeled SW and PC. The SW port should be connected to the Ethernet switch and the PC port used to daisy chain a PC or other Ethernet-connected device. While the SW port supports PoE, the PC port does not, but otherwise the two ports are totally symmetrical.

At present there are no configurable options for either of the external Ethernet Ports.

WAN Interface

The WAN interface on the phone refers to the internal Ethernet switch port that is connected directly to the phone processor. The following setting groups are available.

VLAN

The OBi Phone supports VLAN tagging in compliance with 802.1p/q. If ***WAN Settings – Internet Settings***::**VLANEnable** is enabled, outbound traffic will be tagged according to the parameters **VLANID** and **VLANPriority**. The phone will ignore inbound traffic that does not belong to the same VLAN.

LLDP

The OBi Phone supports LLDP-MED to automatically discover Network Policy (VLAN and DSCP) settings and perform other related handshake functions. This feature is enabled using the parameter ***WAN Settings – Internet Settings***::**LLDP-MED**.

IP Address Assignment

The OBi Phone supports 3 methods of acquiring an IP address assigned to its WAN interface. The method to use is controlled by the parameter ***WAN Settings – Internet Settings***::**AddressingType**, which can have one of the following values:

- `DHCP`: Request address assignment from a DHCP server
- `PPPoE`: Request address assignment from a PPPoE server
- `Static`: Use the statically assigned IP address, subnet mask, and default gateway from the parameters ***WAN Settings – Internet Settings***::**IPAddress**, **SubnetMask**, and **DefaultGateway** respectively

DNS Servers

You can specify up to two DNS servers to be used with the WAN interface in the parameters ***WAN Settings – Internet Settings***::**DNSServer1** and **DNSServer2**. Note that if the DHCP offer includes DNS Servers, the OBi Phone takes up to 16 servers from the list and uses them together with the explicitly configured servers.

WiFi Interface

The OBi Phone supports WiFi. While the OBi2182, OBi2162 and OBi1062 has built-in WiFi hardware, the OBi1022, OBi1032 and OBi2062 can connect via WiFi with an OBiWiFi5G adapter connected to USB Port 1 on the back of the phone (note that you MUST NOT connect OBiWiFi5G to USB Port 2). Note that VLAN and LLDP features are not available on WiFi.

IP Address Assignment

The OBi Phone supports 2 methods of getting an IP address assigned to its WiFi interface. The method to use is controlled by the parameter ***WiFi Settings – Basic Settings***::**AddressingType**, which can have one of the following values:

- `DHCP`: Request address assignment from a DHCP server
- `Static`: Use the statically assigned IP address, subnet mask, and default gateway from the parameters ***WiFi Settings – Internet Settings***::**IPAddress**, **SubnetMask**, and **DefaultGateway** respectively

DNS Servers

You can specify up to two DNS servers to be used with the WiFi interface in the parameters ***WiFi Settings – Internet Settings***::**DNSServer1** and **DNSServer2**. Note that if the DHCP offer includes DNS Servers, the OBi Phone takes up to 16 servers from the list and uses them together with the explicitly configured servers.

DHCP Options

The OBi Phone supports the following DHCP options for both networking interfaces:

- 66
- 150
- 159
- 160
- 161

The options that the phone will try to extract from DHCP offer is a comma separated list of option numbers specified in the parameter ***WAN Settings – DHCP Client Settings***::**ExtraOptions**. Note that the phone will not recognize any option numbers other than the supported ones listed above. You can use the macros `$DHCPOPT66`, `$DHCPOPT150`, `$DHCPOPT159`, `$DHCPOPT160`, and `$DHCPOPT161` to refer to the values of these options in any of the configuration parameters. For example, the default value of ***Auto Provisioning – ITSP Provisioning***::**ConfigURL** is `tftp://$DHCPOPT66/$DM.xml`.

DNS Lookup

The DNS behavior described below applys to both network interfaces.

Lookup Order

In cases where there are multiple DNS servers available, the phone will attempt to resolve a domain name quickly by querying as many DNS servers as necessary. A short delay can be inserted between trying each DNS server sequentially such that the querying will stop as soon as a positive response is received from any of the servers. This desired short delay in seconds can be configured in ***WAN Settings – DNS Control***::**DNSQueryDelay**. When the delay is set to 0, all the DNS servers are queried at the same time. And in cases where there are DNS servers obtained from DHCP and from statically configured values, the order of querying the two groups of servers can be controlled through the parameter ***WAN Settings – DNS Control***::**DNSQueryOrder**. Essentially you can choose to query the statically configured DNS servers first, or the DHCP supplied DNS servers first.

Locally Configured DNS Lookup Table

You may define up to 30 local DNS records in the phone configuration such that the phone will search through these 30 records first before hitting the external DNS services when attempting to resolve a domain name. These records can be A or SRV records. This feature is particularly useful when you want to enable proxy redundancy without using any DNS servers. Note that the only way to provide a list of redundant servers to the phone is through the use DNS A or DNS SRV records.

NTP Servers and Local Time

The phone keeps track of current time by querying NTP servers (using SNTP). Up to 2 NTP servers may be configured using the **NTPServer1** and **NTPServer2** parameters. The local time is determined for the local time zone that is set in the **LocalTimeZone** parameter. The phone queries the NTP servers once per hour to update the current time, which is interpolated by the phone using its own local clock in-between NTP refreshes.

Daylight saving time can be enabled by enabling the option **DaylightSavingTimeEnable**. Daylight saving time is automatically adjusted based on the start and end rules specified in **DaylightSavingTimeStart** and **DaylightSavingTimeEnd** parameters. The amount of time to adjust when daylight saving time is in effect can be configured in the **DaylightSavingTimeDiff** parameter.

There is another mechanism that can be used by the phone to tell time and it is based on SIP signaling. When the phone renews registration with a SIP proxy server, the server may include a Date header in the response to the phone that indicates the current GMT time. The phone will process this time value the way it does with the result from the NTP servers, if ***ITSP Profile X – SIP***::**X_ProcessDateHeader** is enabled.

802.1X Authentication

OBiPhone supports the following 802.1X Authentication Modes:

- Disable
- MD5
- TLS
- TTLS/MSCHAPv2
- PEAP-MSCHAPv2

Authentication Mode is set using the parameter ***WAN Settings – Internet Settings***::**802_1XMode**. Depending on the selected mode, additional authentication parameters may be required according to the following table.

Parameter	Description	(EAP) MD5	(EAP) TLS (1.0)	TTLS/ MSCHAPv2	PEAP-MSCHAPv2
802_1XIdentity	A username. If the value is not needed, it should be set to an empty string	Required	Required	Required	
802_1XPassword	A password or passphrase. If password/passphrase is not needed, this value should be set to an empty string	Required	Required	Required	
802_1XCACertificate	Internal path where the certificate of the CA that signs the server certificate (for 802.1x authentication) is stored		Required	Required	
802_1XClientCertificate	Internal path where the client certificate (for 802.1x authentication) is stored		Required		

Certificates for 802.1X Authentication

As described in the last section, a CA certificate and a client certificate may be required in certain authentication modes. The required certificates must be installed on the device for the authentication to work properly. Note that a dedicated CA certificate and client certificate must be installed on the phone to be used for 802.1x authentication. These certificates must be stored in an internal memory location and the corresponding paths specified in the **802_1XCACertificate** and **802_1XClientCertificate** parameters. All certificates must be in in DER or PEM format. In case of (EAP) TLS, the client certificate file must also include the private key (PEM) file appended to the client certificate (i.e, concatenate the client certificate and private key into a single file to be stored internally).

The default values for the two certificate paths are `${USERDIR}/certs/ca.pem` and `${USERDIR}/certs/client.pem` where `${USERDIR}` represents the root of the user data folders in the Internal drive, which can be accessed from the phone GUI: Main Menu→Settings→Storage→Device (use the < and > keys to select **Internal**)

There are several methods to install these certificate files:

a) From the phone GUI (Main Menu→Settings→Storage→Device), copy the files from a USB flash drive attached to the phone from the GUI and paste them into the Internal drive

b) Package the files together with other Phone Customization Data to upload to the phone via remote provisioning. In this case, the file path should use the macro `${ITSPDIR}` as the root folder to refer to these certificate files. Please read the section on Phone Customization Data for more details

c) Use a OBi IP Phone XML App to install the files into an internal location. The XML APP may be pushed to the phone using SIP/NOTIFY or HTTP/POST method, or pulled by the phone by invoking an Action URL function via a feature key or softkey. Please refer to this article on how to create a Phone XML App.

Feature Keys

A feature key on the phone is one that can be configured to perform one of many different predefined functions, with a corresponding multicolor LED that shows the status of the assigned function instance. There are many feature keys on a OBi1000 series phone; the (Virtual) Line Keys, the Programmable Keys, and the Side Car Keys are all feature keys that may be configured in similar ways. Each feature key may be configured by the phone administrator to perform on of the following functions:

Feature Key Function	Description	Icon
Call Appearance	Make or receive one call, and the key is known as a **Call Key** in this case. You must have an unused (i.e. idle) call key available in order to make or receive a new call. The phone administrator should allocate as many call keys on the phone as the maximum number of concurrent calls it is expected to handle. The VLKW (Virtual Line Key Window) shows nothing but the idle phone icon (shown on the right column) when no active call is assigned to the key. Otherwise the icon changes to reflect the current call state (Call States and Call State Icons are described in the Section *Making and Receiving Calls* and VLKW shows more information about the call, such as the call peer's name and number, if available. VLKW may also change its background color to further reflect the current state. A Call Key may be bound to a voice service and is called a bound call key in that case (otherwise it is known as an unbound call key). A bound call key is used to make/received on the bound voice service only; this is one of the ways for a user to select a specific line to make a call. A call key may be bound to a service that is a Shared line. In that case when no call is on that key, the VLKW information reflects the state of the respective Share Call Appearance (SCA). See the section *Shared line and Share Call Appearances* for more details on this topic. Parameters: - Service: Optional. The service or line to bind the key with - MaxCalls: Number of calls to be managed by this Call Key	No Call  Dialtone  Dialing  Connected  Connected-S Connected-HD  Connected-HDS  Holding  Trying  Peer Ringing  Ringing  Call Ended  SCA: Error  SCA: Idle  SCA: Line Seized  SCA: Trying  SCA: Proceeding  SCA: Connected

		SCA: Call Parked SCA: Holding SCA: Private Holding
Line Monitor	Monitor a Line (i.e. a voice service installed on the phone). The Line events that are monitored include: • Ringing: at least one incoming call • Holding: at least one call holding • In Use: at least one active call • Idle: no calls VLKW shows the monitored service name and account user name (usually same as the account DID number or extension). This function must be bound to the specific voice service that it monitors. Parameters: - Service: Required. The service or line to monitor	Idle Ringing In Use Holding
BLF (Busy Lamp Field)	Monitor the call state of another extension. A BLF key must be bound to a service (as configured by the phone administrator). The call events that are monitored include: - Ringing: at least one incoming call - Holding: at least one call holding - Busy: at least one active call - Idle: no calls - Call parked (against the monitored extension) VLKW shows the bound service name and the monitored extension (or DID number, or account user name). This function must be bound to a specific voice service. Parameters: - Service: Required. The service that provides the monitoring function - Number: Required. The extension (on the specified service) to monitor; it may contain multiple attributes ○ ptt – (no value). Make the call a Pust-to-talk when calling the monitored extension ○ spd – An alternative number to call when calling the monitored extension ○ bx – (no value). Enable one-touch blind transfer behavior (i.e. highlight a call that is eiher connected or holding on screen and press this key to blind transfer the call to this extension)	Idle Busy Call Parked Ringing Offline Holding
Call Park Monitor	Monitor the call park status of a single orbit in a parking lot. Per default LED setting, the key's LED is turned off when no call is parked in that orbit, or otherwise solid red. When the orbit is not occupied, press the CPM key once to park the highlighted call on screen onto that orbit (the call must be in the Connected or Holding state to be parked). When the orbit is occupied, press the CPM key to retrieve the call as an additional call on the current phone. Parameters: - Service: Required. The service that provides the parking lot orbit: parking and monitoring function - Number: Required. The parking lot orbit to park and monitor	

Presence Monitor	Monitor the presence/status of one buddy in a Buddy List. It also serves as speed dial to that buddy Parameters: - Service: Required. The service that provides the XMPP service for this function - Number: Required. The JID of the buddy in the buddy list to monitor. Only the userid portion of the JID is needed. You may specify .. at the end such that a partial match of the JID is enough. For example, if the JID of the buddy is: *abcd-12345@gmail.com*, you may specify `abcd..` for the Number field to monitor this buddy, provided there is no other JID in the buddy list that starts with abcd.	Online Offline Extended Away Away Do Not Disturb Unknown
Speed Dial	If a number has not been assigned, VLKW shows no textual information. Otherwise it shows the speed dial number configured and, if the speed dial is bound to a service, the name of the service that it is bound to. If the speed dial has a display name configured, the VLIW shows the display name (such as `John Seymour`) instead of the assigned number. Parameters: - Service: Optional. The suggested service the make the call with - Number: Required. The number to call. It may include the following attributes: ○ `ptt` (with no value): make the key a Push-To-Talk key. ○ `bx` – (with no value). Enable one-touch blind transfer behavior (i.e. highlight a call that is eiher connected or holding on screen and press this key to blind transfer the call to this speed dial number ○ `send=`{digit-codes} to automatically dial some digits after the call, where {digit-codes} is a sequence of the following case-sensitve codes: ▪ 0-9,*,#,a,b,c,d – The DTMF digit to send to the peer. Each digit is sent with 100 ms on and 100 ms off ▪ S – Pause for 3s ▪ s – Pause for 1s ▪ U”{prompt}” – Prompt the user to enter one more digits manually on the phone with the given {prompt} shown on the screen. User then press “OK” soft key to continue ▪ A – Wait for the called party to answer before continuing Note that the phones starting executing the first code in {digits} when the call receives early media or when the call is answered, whichever happens first. For example: `send=Ass1234U”Enter Passcode”5678` It is aceeptable to have the send attribute specified in a speed dial without a number (e.g. Number=`;send=1234#`). In this case the speed dial can be used to send out the list of digits on a connected call. Note that if a voice service is specified for the speed dial either in the Service Field or the given number is a full number (i.e. one that includes voice service information), the phone does not apply digitmap before calling the speed dial. Hence any *codes in the number are not processed and the result may not be desirable. You can include the *codes to be processed by the phone by enclosing them in a pair of [...] at the beginning of the number field. For example: you may set the number field to `[*96]SP1(2113);ptt`. The phone then inteprets *96 locally (to request auto-answer on the called party) and makes the call to 2113 using SP1 service.	

	Using Speed Dial to store a Feature Access Code prefix There are many applications where a function can be invoked from the Service Provider by calling a number that is a feature access code prefix followed by a target number. For example, the prefix *48 followed by the target extension such as 1002 may invoke from the Service Provider the call-monitoring function on extension 1002. For this the user may just dial *481002 directly. For better UX, however, you may also store *48 in a speed dail and label it as Monitor on the GUI. To mark the speed dial as a prefix, you append 2 or more dots to the number, such as *48… When user presses the Monitor speed dial, the phone shows a dial box on screen with *48 entered. User may then continue to dial the target extension, or press another Speed Dial or BLF that is the target extension to monitor.	
Do Not Disturb On/Off	Turn on/off the **Do Not Disturb** feature. If the function is bound to a specific voice service, it is applied to incoming calls on that service only. Otherwise, it is applied system wide to all incoming calls regardless which service the calls are coming from). Parameters: - Service: Optional. The service to apply this feature on. If no service is specified, the feature applies to all calls the phone	on off
Do Not Ring On/Off	Turn on/off the **Do Not Ring** feature. When the feature is enabled, incoming calls comes through like normally but phone does not play audible ring (call waiting tone however will be played during call-waiting). This function cannot bound to any specific voice service; it is applied system wide to all incoming calls regardless which service the calls are coming from. Parameters: None	on off
Block Anonymous Callers On/Off	Turn on/off the **Block Anonymous Caller** feature. If the function is bound to a specific service, it is applied to incoming calls on that service only. Otherwise, it is applied system wide to all incoming call regardless which service the calls are coming from. If this feature is enabled, the phone rejects all incoming calls with Caller ID (name/number) hidden (a.k.a. blocked). Parameters: - Service: Optional. The service to apply this feature on. If no service is specified, the feature applies to all calls on the phone.	on off
Block Outgoing Caller ID On/Off	Turn on/off the **Block Caller ID** feature. If the function is bound to a specific service, it is applied to outgoing calls on that service only. Otherwise, it is applied system wide to all outgoing calls regardless which service is used for the calls. If this feature is turned on, the phone will attempt to hide user's caller ID on outbound calls so the called party cannot see who's calling. Parameters: - Service: Optional. The service to apply this feature on. If no service is specified, the feature applies to all calls on the phone.	on off
Call Forward All On/Off	Turn on/off the **Call Forward All Calls** feature. If the function is bound to a specific service, it is applied to incoming calls on that service only. Otherwise, it is applied system wide to all incoming calls regardless which service the calls are coming from. Parameters: - Service: Optional. The service to apply this feature on. If no service is specified, the feature applies to all calls on the phone.	on off

Auto Answer On/Off	Turn on/off the **Auto Answer (Intercom Calls)** feature. Normally this feature is turned on so that incoming intercom calls can be automatically answered by the phone by turning on the speakerphone or headset. If this feature is turned off, incoming intercom calls will be treated as regular calls and the phone will ring normally. This function cannot be bound to any specific voice service; it is applied system wide to all incoming calls regardless which service the calls are coming from. Parameters: None	on off
Call Waiting On/Off	Turn on/off the **Call Waiting** feature. Normally this feature is enabled such the user can accept more incoming calls while he is already on a call. If this feature is turned off, all incoming calls will be rejected as busy when user is on a call. This function cannot be bound to any specific voice service. Parameters: None	on off
Message Status (Monitor Voicemail Status)	Monitor the number of messages in a mail box. The function must be bound to a SP service that has the MWI (Message Waiting Indication) feature enabled. The VLKW shows if and how many new messages are available in the mailbox. Parameters: - Service: Required. The SP service that provides the Voicemail service. - Number: Optional. The number to call (to check voicemail). It may include the following attributes: o `mbox=`{*mailbox-id*} where {*mailbox-id*} is 3rd party mailbox to monitor, if it is not the same as the main mailbox for the given SP Service. On the other hand, do not specify this attribute if you are monitoring the main mailabox for the given SP Service	messages no messages
Hold	Hold all calls that are in the Connected State. VLKW shows how many calls are currently in Holding State. The LED turns red if there is a least one call in the Holding State, or turned off otherwise. This function cannot be bound to any specific voice service. Parameters: None	
Add to Conference	Add all calls that are in the Holding State to the current conversation. VLKW shows how many calls are in the Holding State. The LED turns red is there is at least one call in the Holding state, or turned off otherwise. This function cannot be bound to any specific voice service. Parameters: None	
Join/Leave Page Group 1	Join/Leave Page Group 1. If the user joins the group, then the phone automatically turns on the speaker when anyone in the group starts a page. User may also start the page by pressing down the feature key; this is known as PTT or Push-to-talk (otherwise the user is just listening). If allowed by the phone admin, user may also "Clamp On" the feature key to talk continuously without needing to hold down the key. This function cannot be bound to any specific voice service. Parameters: None	left joined
Join/Leave Page Group 2.	Same as the last item except this one applies to Page Group 2	left joined

Change ACD Agent State	Change/monitor an ACD (or Call-Center) Agent State to one of the following values: - Available (to take new calls) - Unavailable (to take new calls) - Signed Off - Wrapping Up (the last call) ACD stands for Automated Call Distribution is the primary way a call-center distributes calls among a number of agents working for the call-center. The ACD controller should only send incoming call to an agent whose state is "Available". An agent may sign off when he's done for the day, or unavailable when he's talking a break, or wrap up if he has to do some paper work for the last customer call before he can take another call. While "Signed Off", agent presses the key once to sign on and becomes "Available". While "Available", agent presses the key once to become "Unavailable". While "Unavailable" or "Wrapping Up", agent presses the key once to become "Available". Note that agent cannot change state to "Signed Off" or "Wrapping Up" directly by pressing the feature key. To change to these states, agent must use the corresponding feature key menu item from the GUI (invoked by pressing and holding down the feature key), or some other means provided by the Soft Switch, such as a web portal. This function must be bound to a specific voice service. The ACD agent handles calls on the bound service only w.r.t. to the underlying Call Center. The Call Center is not aware of calls the agent makes or receives with other voice services installed on the phone. Parameters: - Service: Required. The service that provides the ACD Agent function.	 signed off  available  wrapping up  unavailable
Guest User Login/Logout.	This feature is also known as Hoteling on some Soft Switch. The phone may be set up to be used temporarily by a guest, such as a visiting employee or temp worker. The guest can press the key and enter a user-id and password to log in and start using the guest phone for his own extension temporarily (until he logs out or the server logs him out remotely). This function must be bound to a specific voice service (that supports this feature). Parameters: - Service: Required. The service that provides the Guest Login function.	 guest logged in  guest logged out
Enter Disposition Code for the last call.	Enter a **Disposition Code** for the last call. This is used by a Call Center agent to enter a disposition code for the last customer call. This function must be bound to a specific voice service (that supports this feature). Parameters: - Service: Required. The service that provides the Dispostion Code function	
Next Tab	Pressing this key to switch to the next VLK Tab or cycle back to VLK Tab 1. This function cannot bound to any specific voice service. Parameters: None Note that the multiple Line Key Tabs feature can be disabled also by disabling the option ***User Preferences***::**LineKeyTabs**. In that case the "Next Tab" function does nothing.	

Transfer	Invoke the call transfer function on the currently highlighted call on the screen when the Calls App is at the top of the display stack. The call must be in a transferrable state, that is, Holding or Connected State. Parameters: - Service: Optional. The suggested service to use to make the call to the transfer target, if the Number parameter is also specified - Number: Optional. The number of the transfer target to call. User will not be prompted to enter the transfer target number if this parameter is specified	
Blind Transfer	Invoke the blind call transfer function on the currently highlighted call on the screen when the Calls App is at the top of the display stack. The call must be in a transferrable state, that is, Holding or Connected State. Parameters: - Number: Optional. The number of the transfer target to transfer the call to. User will not be prompted to enter the transfer target number if this parameter is specified	
Blind Transfer 2	Same as the **Transfer** function w.r.t. collecting the target number from the user, but a blind transfer is performed after the target number is collected. In contrast to **Blind Transfer**, **Blind Transfer 2** plays dial tone, applies digit map, and would time out as user enters the target number	
Action URL	Launch an OBiPhone XML Application at the given URL Parameters: - Number: Required. The URL of the OBiPhone XML Application to run - Name: A name to identify the application on the phone screen	

Every feature key has a corresponding **Feature Key Item** that can be viewed from the phone GUI. As each feature key belongs to one of four *feature key groups* (Line Keys, Programmable Keys, Side Car 1 Keys, and Side Car 2 Keys), feature key items are similarly grouped and ordered in their corresponding feature key item lists on screen, where there is one feature key item list for each of the four feature key groups. The easiest way to bring up the Feature Key Item is by pressing and holding down the corresponding feature key until the feature key item list appears on the screen with the pressed key item highlighted; the normal function that is assigned to the feature key is not be triggered in this case. With the Feature Key Item, user can view additional information about the assigned function and invoke other operations (via soft key options for example) that are not accessible by a simple key press of the feature key itself. You can also access the feature key item lists by going through the Settings menu on the phone GUI.

Line Keys and Virtual Line Keys

There are 6 physical Line Keys (LKs) on the OBi1062/2182 (3 on the OBi1032/2062/2162 and 5 on the OBi1022). Four (4) “tabbed” pages (2 on the OBi1022) of Virtual Line Key (VLK) are defined such that under each tab there are 6 corresponding virtual line keys (3 on OBi1032 and 5 on the OBi1022). Hence a total of 24 VLKs (12 on OBi1032 and 10 on OBi1022) are available per phone. The VLK tabs are numbered 1–4 (1–2 on the OBi1022), where one and only one VLK tab is visible on the screen at any time, the 4 Tab icons (2 on the OBi1022) on the status bar of the screen makes it clear which tab of VLKs is currently on the top. The VLKs themselves are numbered as 1 – 24 (1–12 on OBi1032/2062/2162 and 1–10 on OBi1022) such that LK1 – LK6 are mapped to VLK1 – VLK6 under Tab 1, to VLK7 – VLK12 under Tab 2, to VLK13 – VLK18 under Tab 3, and to VLK19 – VLK24 under Tab 4. Similarly, on the OBi1032/2062/2162, LK1 – LK3 are mapped to VLK1 – VLK3 under Tab 1, to VLK4 – VLK6 under Tab 2, to VLK7 – VLK9 under Tab 3, and to VLK10 – VLK12 under Tab 4, and on the OBi1022, LK1 – LK5 are mapped to VLK1 – VLK5 under Tab 1, and to VLK6 – VLK10 under Tab 2. The respective LK LED reflects the status of the mapped VLK under the visible tab only. To see the LED pattern corresponding to a VLK that is not visible, user must “switch to the tab” that contains the

said VLK. In order to switch to the next tab, a feature key must be assigned the function “Next Tab” such that pressing that key advances to the next VLK tab or recycles to tab 1.

As a feature key, a VLK is different from the other keys in that it also has an associated window area on the screen along the right edge next to the physical Line Key. This is called a **Virtual Line Key Window (VLKW)**. Only the VLKW of the VLKs that are on the current VLK Page are visible. A VLKW provides additional feature key function dependent information. For example, if the key is assigned the "call" function, its VLKW may show the call peer's name or number during a call, with an icon or background color that reflects the call state.

Left Line Keys and Virtual Left Line Keys

Available on the OBi2000 Series Only.

There are additional 6 physical Left Line Keys (LLKs) on the left side of the screen on the OBi2182 (3 on the OBi2062/2162), that are symmetrically positioned versus the 6 physical (Right) Line Keys. Like the (Right) Line Keys, there are 4 tabs of Virtual Left Line Keys (VLLKs), thus providing additional 24 VLLKs (12 on the OBi2062/2162) on these phones. The programming and functionality of the VLLKs are the same as the LLKs.

Programmable Keys

There are eight (8) Programmable Keys on the OBi1032/1062 and one (1) on the OBi1022. These are called PK1 – PK8.

Programmable Hard Keys

There are no Programmable Keys in the OBi2000 series. Instead they have 9 additioanl Hard Keys that are wired to the most commonly used features with function-specific icons printed on the physical keys . Theoretically, though not recommended, one can re-program these additional Hard Keys to provide other functions in general, like the Programmable Keys in OBi1000 series. The recommended and default functions of these 9 Hard Keys are:

1. **Phone Book**

   - Default **Function** = `Action URL`
   - Default **Number** = `phone://phone-book`

   The default setup of this key to access the local Phone Book (a built-in Phone App). You can map this key to invoke instead the LADP directiory by setting

   Number = `phone://netdir?service=LDAP`

   Or the Network Directory configured under one of the SP*n* service by setting

   Number = `phone://netdir?service=SP`*n*

2. **Conference**

   - Default **Function** = `Conference`

   The default setup of this key provides similar action the **conf** soft key would, if the highlighted call is in the Connected or Holding state. After the user dials the new conferee target number, where the screen would show the **conf.now** soft key, press this key triggers the same action of the **conf.now** soft key instead.

   A valid alternative to this behavior is to change **Function** to `Add To Conference` when pressing the key would add ALL calls currently in the holding state to a conference call.

3. **Transfer**

   - Default **Function** = `Transfer`

The default setup of this key provides similar action the **transfer** soft key would, if the highlighted call is in the Connected or Holding state. After the user dials the transfer target number, where the screen would show the **xfer.now** soft key, press this key triggers the same action of the **xfer.now** soft key instead.

Two valid alternatives to this behavior is to change **Function** to `Blind Transfer` or `Blind Transfer 2` when pressing the key would provide corresponding behavior, respectively

4. Hold

- Default **Function** = `Hold`

The default setup lets the user put on hold all the calls that currently in the connected state by pressing this key.

5. Do Not Disturb

- Default **Function** = `Do Not Disturb`
- Default **Service** = {*blank*}

The default setup lets user turn on/off Do Not Disturb for the entire phone, locally. One may change the Service to one of the SP*n* Service instead. And if the selected SP*n* service has the network-provided version of DND enabled, this key controls the on/off of the netowork-provided version (instead of the version provided by the phone locally). The default configuration sets the LED to solid red when the underlying DND setting is ON, or off otherwise

6. Call Forward

- Default **Function** = `Call Forward`
- Default **Service** = {*blank*}

The default setup lets user turn on/off Call Forward Unconditional (a.k.a. Call Forward All or Call Forward Always) for the entire phone, locally. One may change the Service to one of the SP*n* Service instead. And if the selected SP*n* service has the network-provided version of Call Forward Always enabled, this key controls the on/off of the netowork-provided version (instead of the version provided by the phone locally). The default configuration sets the LED to solid red when the underlying Call Forward Always setting is ON, or off otherwise.

7. Voicemail

- Default **Function** = `Message Status`

You should set up the **Number** and **Service** parameters of this key to access voicemail.

8. Next Tab

- Default **Function** = `Next Tab`

The default setup provides the function of changing the (right) line key tab. We do not recommend changing the function of this key.

9. **Left Next Tab**

   - Default **Function** = `Left Next Tab`

   The default setup provides the function of changing the left line key tab. We do not recommend changing the function of this key.

Side Car Keys

Up to two side cars can be connected to the phone (none for the OBi1022 and OBi2000 Series) by daisy chaining. Each side car presents 16 additional feature keys. The side car attached directly to the phone is referred to as Side Car 1; the side car attached to Side Car 1 is referred to as Side Car 2.The feature keys are referred to as SC1K1 – SC1K16 and SC2K1 – SC2K16 respectively.

Note: The OBi2000 Series do not support any side car at this time

Feature Key Configuration Parameters

Each feature key can be configured independently. The configuration of each feature key comprises of the following set of parameters:

- **Function**: Select the function to assign to this feature key
- **Service**: The service to bind the key to. Required for ACD Sign On/Off, Busy Lamp Field, Call Park Monitor, Disposition Code, Hoteling, Exec Filter On/Off, Exec Assistant, Line Monitor, Message Status, Presence Monitor, and Security Class. Optional for Block Anonymous Call, Block Caller ID, Call Appearance, Call Forward, Do Not Disturb, Transfer, and Speed Dial. Not used otherwise.
- **Number**: Required for BLF, Call Park Monitor, Presence Monitor, and Speed Dial. Optional for Transfer and Blind Transfer
- **Name**: Optional for all functions. It is used as a nickname to refer to the entity specified in the **Number** parameter.
- **MaxCalls**: Maximum number of calls to overload on the key. This is only applicable if the function is Call Appearance (therefore, this parameter is not available under Programmable Keys and Side Car Keys).
- **Group**: Reserved for future use.

It is very common to have multiple feature keys defined with the same function, such as Call Appearance and Speed Dial. It is not advisable to have a particular monitor function with the same monitored entity configured on more than one key. For example do not assign more than one BLF key to monitor the same extension, or assign more than one Presence Monitor key to monitor the same buddy, or assign more than one Message Status key to monitor the same mailbox, etc. The reason is that the phone may only update just one of keys when the status of the monitored entity is changed.

Highlights of Feature Key Functions

This section highlights the most commonly used feature key functions to introduce some definitions that are used frequently through out the document. More details on each function can be found in other sections.

Call Keys

Each VLK is a feature key that can be assigned the "call" function, and is also referred to as a Call Feature Key, or simply Call Key. Each call carried out on the phone must be assigned a call key. That is, you will need at least one Call Key to make or receive a call. Each Call Key can hold exactly one call. Usually there are at least a few Call Keys defined on the

phone to handle multiple simultaneous calls scenarios such as call waiting and conference calls. If there is a new incoming call but no more Call Keys to assign the call to, it will receive the busy treatment.

A Call Key may be configured as *bound* to a specific Voice Service account (such as SP1) installed on the phone, or *unbound* to any service. A Bound Call Key is used to handle calls on the bound service account only, while an Unbound Call Key can handle calls on any service account. For incoming calls, the phone automatically assigns the call to an open Call Key. It first attempts to find a Call Key that is bound to the service account where the incoming call is on. If none is found, it then tries to look for an unbound call key to assign the call to. If none is found, the call is rejected with busy treatment.

A Call Key may be "overloaded" with up to four calls. The number of calls to overload on a Call Key is controlled by the **MaxCall** parameter. When the key is overloaded, the LED pattern and the key display reflects the states of only one of the calls at any time; this call is referred to as the call-in-focus. The call-in-focus is selected by the phone automatically according to the following:

- The latest Ringing call; if multiple available, the latest one
- The latest Holding call
- The latest Call in other states

A user can also force a call to be the call-in-focus from the GUI. The operation triggered by pressing the Call Key applies to the call-in-focus only according to its current state.

Line Monitor Keys

Not to be confused with a line key, a line monitor key monitors the status a line, whether it is up or down, has calls ringing, holding or connected. Pressing the key may result in:

- Answer an incoming call if there is one. If there are more than one, answer the oldest one
- Start dial tone, if there are spare capacity

Speed Dial Keys

The Speed Dial function lets the user configure a speed dial number from the phone GUI. User can press and hold down the speed dial key until the feature key item shows up on the screen. From there the user can configure the speed dial details. The service to use for calling with the speed dial may also be configured. This function supports the Push to Talk PTT option.

Note: There are in addition to the spee dial feature keys 99 configurable speed dial numbers that may be invoked by dialing the corresponding 1-2-digit speed dial code (that is, 1 – 99) like a regular number. These 99 speed dial storages are referred to as the "***Speed Dials 99***" feature that is entirely independ of the speed dial feature keys discussed here.

BLF Keys

BLF or Busy Lamp Field is described in detail in a later section. A BLF key is used to monitor the call status of another extension. This feature operates in the context of an SP service. In many cases the BLF key also acts as a speed dial key to call or transfer a call to the monitored extension. This key also supports PTT when it is used to call the monitored extension. Note that the call or call transfer to the monitored extension will use the same underlying SP service.

Call Park Monitor Keys

CPM or Call Park Monitor is very similar to BLF, except each key monitors a parking lot orbit instead of another user extension. It only indicates the status whether a call is parked on the orbit or not. Pressing the key once to either park the highlighted call on screen to the orbit if it is not occupied, or retrieve the call from the orbit if it is currently occupied.

Presence Monitor

This function is used together with a Buddy List. You can configure a feature key to monitor the presence/status of a buddy in a buddy list. You can also use this key as a speed dial to that buddy. Note that the Buddy List feature operates in the context of an SP service. The call to the buddy by pressing a presence monitor key will use the same underlying SP service to make the call. The PTT option is NOT available with this key when calling the buddy.

Group Page Keys

OBi1000 supports two (multicast) page groups called Page Group 1 and Page Group 2 respectively. A feature key must be set up with the function Page Group 1 or Page Group 2 in order to use the respective page group. Paging is one-way; incoming audio will not be played by the phone when the user is talking. The configuration of each page group has a Push-To-Talk option which can be enabled such that user must press the page key to talk. The key also lets the user the join or leave the group with one key press. For example when the user does not want to be bothered by incoming page, he can temporarily leave the group. The LED color also reflects the current group-joining status as a reminder to the user.

Input Methods

Phone Keypad

You can input digits or text directly from the phone keypad. There are three input modes:

- Numeric (123): Each keypress input the corresponding digit 0–9, *, #
- Alphanumeric (abc): Each keypress invokes a choice of ASCII charaters where the user can select the character to input
- Capitalized Alphanumeric (Abc): Same as alphanumeric mode except the ASCII character selection has default set to a capital letter
- IPv4 Address (IPv4): Same as numeric but with * maps to a dot (.) and # maps to a colon (:), for entering an IPv4 address

Users can change between input modes by pressing the `mode` soft key (by default, label is “Switch Mode”)

USB Keyboard

You can connect a USB keyboard to a USB port at the back of the phone and use it to enter standard ASCII text. The icon  or  will appear on the notification area of the screen when a keyboard is connected, where the latter icon indicates caps lock is on. For input box that shows an input mode switch, just use the ‘123’ mode when entering text from the USB keyboard to allow normal text input (without selection of an input symbol from a table as in Abc/abc mode). Note that USB Keyboard and Phone Keypad can be used together at the same time.

In addition, the following shortcuts can be used to emulate other phone key presses or phone action from a USB keyboard:

USB Keybaord Key	Phone Key	USB Keyboad Key	Phone Action
ESC	Cancel/Back	Tab	Focus next menu item on the screen
Enter	OK/Select	Ctrl Tab	Next Line Key Page
Home	Home	Ctrl Alt Del	Reboot
Up/Down	Up/Down Navigation	PgUp	Set cursor to 1st input character
Left/Right	Left/Right Navigation	PgDn	Set cursor to the last input character
F1/F2/F3/F4	Soft Key 1/2/3/4	End	Set cursor to the last input character of the current row
Ctrl F1/F2/F3/F4//F5/F6	Line Key 1/2/3//4/5/6	Backspace	Backspace
Ctrl s/S	Speaker	Del	Backspace
Ctrl h/H	Headset		
Ctrl m/M	Mute		
Ctrl 1/2/3/4/5/6/7/8	Programmable Key 1/2/3/4/5/6/7/8		
Ctrl +	Vol Up		
Ctrl -	Vol Down		

Voice Services

A Voice Service, also known as a Line or Trunk, is an individual user account with an ITSP. The following voice services can be configured on an OBi1000 IP Port:

- SP1 (SIP or Google Voice)
- SP2 (SIP or Google Voice)
- SP3 (SIP or Google Voice)
- SP4 (SIP or Google Voice)
- SP5 (SIP or Google Voice)
- SP6 (SIP or Google Voice)
- OBiTALK
- OBiBluetooth

An SP*n* service can be a generic SIP voice service or a Google Voice service. Examples of SIP/SP service: An extension from a PBX, a subscriber account with a service provider. Every SP service user account requires a username; a password is often required also for authentication. The service provider assigns an extension number or DID number to the user account; the assigned number may or may not be the same as the username of the account.

OBiTALK is a built-in service provided and managed by Obihai. It can be used for technical support as well as free device-to-device calling among OBi devices. OBiBluetooth can be thought of as an internal service that is made available by the user pairing/connecting to a mobile phone via Bluetooth; this Bluetooth connection serves as a gateway for the phone to access the mobile phone service as another trunk.

More information is available about each type of voice service later in this document.

ITSP Profiles

The configuration of an SP service is divided into two parts: A Service Provider part and a Service Subscriber Part. The Service Provider part comprises of parameters that are common to all service subscriber accounts from that service provider. The Service Subscriber part comprises of parameters that may vary for each specific subscriber account from the service provider.

On an OBi device, each Service Provider part is known as an ITSP Profile that has its own parameter groups. Up to 6 ITSP Profiles can be defined in a phone configuration and are referred to as ITSP Profile A – ITSP Profile F . The Service Subscriber part is known as an SP Service, which includes the parameter ***SPn***::**X_ServProvProfile** that binds the SP service with an ITSP Profile. By default, the ***SPn***::***X_ServProvProfile*** parameter for all SP services points to ITSP Profile `A`. Suppose you want to use two different service providers with the phone and configures the settings for them in ITSP Profile A and ITSP Profile B respectively.

A common mistake is not to set the **X_ServProvProfile** parameter correctly to point to the corresponding ITSP Profile as intended.

Overview of Common Trunk Configuration

Trunks of every kind share some common characteristics. This section outlines some of them that are commonly needs to be configured.

Service Enable

Before a trunk can be put into service, it must be enabled in the phone configuration. There is an **Enable** parameter for each service and this parameter is checked by default.

Service Account Credentials

Credentials are required for all SP services only; they are neither required or available for OBiTALK and OBiBluetooth services. At the minimum, a username for the SP service account must be configured with the parameter:

***SPn Service- Service Credentials*::AuthUserName**

If a password is also required for authentication to the server, put it into:

***SPn Service- Service Credentials*::AuthPassword**

In a less common situation where the username used for SIP authentication is different from the account username, this can be achieved by setting the account username in ***SPn Service- Service Credentials*::URI** and the different username for authentication only in **AuthUserName**. In other words, **AuthUserName** MUST be specified and, if **URI** is not specified, it is used for both as the account username and for SIP authentication. Note also that if **AuthUserName** is not specified, the phone considers the service as disabled also.

Servers

The equipment operated at the service provider side can be generically referred to as the *Servers*. For SIP/SP services, we can call them SIP Servers. For the Google Voice service, we can call them Google Voice Servers. And for the OBiTALK service, we can call them OBiTALK Servers. There are no external servers required for the OBiBluetooth service; one can think of the server as built into the phone software in this case.

Another commonly used term for the server equipment is *Soft-Switch*. A soft-switch typically offers a lot of extra business productive/collaboration features in addition to basic phone services. In fact, there are a few popular open-source and commercial soft-switch implementations such as BroadSoft, MetaSwitch, and FreePBX, that the OBi1000 phones fully support.

No matter what technology is used in the service provider side equipment, it must be provisioned into the phone configuration as a domain name or an IP address; a port number is also required if the server is not listening at the standard port (5060). Note that since Google Voice and OBiTALK servers are already known by the phone, there is no need to configure the server domains for these services. For other SIP/SP services, the proxy server must be configured in the parameter: ***ITSP Profile X – SIP*::ProxyServer**. If the listening port is non-standard, configure the correct value in ***ITSP Profile X – SIP*::ProxyServerPort**. The **OutboundProxy** parameter in the same parameter group is often needed when the device-facing server is a Session Border Controller (SBC). Similarly if the outbound proxy is not listening at the standard port, configure the correct port value in **OutboundProxyPort**. On the other hand, the **RegistrarServer** and **RegistrarServerPort** parameters are rarely needed; the phone assumes the SIP Registrar is the same as the SIP Proxy Server if they are not specified separately.

SIP Transport refers to the transport protocol to use to exchange SIP messages with the server: UDP, TCP, and TLS are supported by the phone. The transport protocol is configured by the ***ITSP Profile X – SIP*::ProxyServerTransport** parameter. For TCP/TLS, the phone must start a TCP/TLS connection with the **ProxyServer** and all subsequent SIP messages must be exchanged using the SAME connection. If **OutboundProxy** is specified, the phone will start the TCP/TLS connection with the **OutboundProxy** instead. With the **OutboundProxyTransport** parameter, it is possible to choose a different transport to be used with the **OutboundProxy** and with the **ProxyServer**.

Trunk Capacity

This refers to the maximum number of simultaneous calls that is allowed on the trunk. For OBiBluetooth service, this value is always `1` and is not configurable. For other services, this value can be set in the parameter **MaxSessions** for each service. The default value is 2 for all services. For OBiTALK service, the maximum value is 4. For Google Voice services, the maximum value is 2. For other SIP/SP services, the value must be set to no larger than the maximum number of simultaneous calls allowed by the service provider.

Basic Incoming Call Handling

For each incoming call arriving at the phone from a specific trunk, the phone handles it in the following order:

- Ignore/reject the call if the trunk is not enabled, else
- Forward the call if native per-line *Call Forward Unconditional* feature is enabled on the service, else
- Apply *Busy Treatment* to the call if native per-line *Do Not Disturb* feature is enabled on the service, else
- Apply *Busy Treatment* to the call if the number of existing calls on the trunk already reaches the limit set for the service, else
- Apply the rules in the **InboundCallRoute** parameter of the service to determine where to send the call to. A common destination for an incoming call is `ph` (that is to ring the phone)

Notes:

- Busy Treatment refers to whether to reject/ignore the call or apply native per-line Call Forward On Busy if the feature is enabled on the service
- When it comes to reject/ignore a call, the decision whether to reject or ignore is based on the service. For OBiBluetooth, the call is ignored so it will continue to ring on the connected mobile phone. For other services, the call is rejected
- **InboundCallRoute** can be configured to let the phone do complex call handling, for example:
  - Ring the phone, the AA, and one or more cell phone number(s) via SP*n* simultaneously; whoever answers first takes the call
  - If the caller number ends in 4281234 or 3357, ring the AA and a cell phone number simultaneously; otherwise ring the phone only

More information on this parameter can be found in the *Call Routing* section.

Basic Outgoing Call Handling

The mechanics of selecting a trunk for outgoing call is the subject of a later section. When the trunk receives a number to call, it will make the call if the trunk is enabled and up and it has not yet reached full capacity. Otherwise, the trunk will fail the outgoing call attempt.

It should be noted that although there is a **DigitMap** parameter available per service, it is however not used by the trunk to validate the number to call (the validation is done at a higher level before the call attempt is routed to the trunk for execution). The per-line **DigitMap** is used as a reference in other **DigitMap** parameters (usually in ***Phone Settings***::**DigitMap**) and in trunk selection from within a trunk group. The latter is more relevant to the trunk itself: When a trunk group has been determined as the destination of an outgoing call, the phone selects the trunk from the group by taking into account whether the number to call is valid against the rules in the **DigitMap** of the trunk.

Specification of Target Phone Numbers

There are places within the OBi configuration that specify a target phone number. For example, a speed dial number or a call forward number. Here we define two formats to specify a target phone number: A ***Short Number*** where only the number itself is specified, such as `3231234` or `14089993312`, and a ***Full Number*** where the number and the service to use the number are both specified, such as `pp(ob222222222)` or `sp3(14089993312)`. The case-insensitive service name to use for each service in full number specifications is:

- `sp`*n* for SP*n* Service for *n* = 1 – 6
- `pp` for OBiTALK Service
- `bt` for OBiBluetooth Service

When only a short number is specified, the phone determines the service to use, where necessary, by going through normal digit map and call routing processing, as explained later in this document. When a full number is specified, the phone uses the number and service as specified without any modification.

SIP/SP Service

Up to **6** SIP/SP service accounts can be configured on the OBi Phone. For the purpose of this and other OBi documents and web pages, including the OBiTALK web portal, the term ITSP is used generically to describe the logical entity providing the SIP Trunk service to the OBi. ITSP stands for Internet Telephony Service Provider. When the OBi1000 is used with an IP PBX for instance, it should be understood that the ITSP refers to the IP PBX in this context.

Each ITSP configuration is grouped together as an ITSP Profile. There are **6** ITSP profiles available and we refer to them as ITSP Profile A, B, C, D, E, and F respectively. The SP service account specifics on the other hand are grouped under the heading SP*n* Service, where *n* = 1, 2,…, 6. An ITSP Profile includes such parameters as **ProxyServer**, **OutboundProxy**, and **DigitMap**, but does not include account specific parameters. An SP Service includes account specific parameters such as **AuthUserName** (usually but not necessarily the same as the phone number of the account), **AuthPassword**, **CallerIDName**, and **X_ServProfile** (which ITSP Profile to apply the ITSP specific parameters from). If both SP Services use the same ITSP, it is usually possible to configure just one ITSP Profile with both SP Services referring to the same profile. However if abstraction of an ITSP Profile is not sufficient to cover a particular ITSP, it is perfectly alright to configure multiple ITSP profiles for the same ITSP and have each individual SP service using that ITSP point to a different ITSP profile.

The SP*n* Service using ITSP Profile *X* is considered as *enabled* by the phone only if at least the following parameters are set:

> ***ITSP Porfile X – SIP*::ProxyServer** = *Not Blank*
>
> ***SPn Service*::Enabled** = `true` (or *checked* on the device web page)
>
> ***SPn Service*::AuthUsername** = *Not Blank*

Where *X* = A, B, C, D, E, or F, and *n* = 1, 2, …, or 6. Otherwise the phone considers the service disabled.

SIP Registration

Device can be setup to periodically register with the **ProxyServer** or the **RegistrarServer** by enabling the parameter ***SPn Service*::X_RegisterEnable**. **ProxyServer** and **RegistrarServer** could be different, although they are rarely so in practice. **ProxyServer** is a required parameter that must be configured on the OBi device, while **RegistrarServer** is optional and is assumed to be the same as the **ProxyServer** if not specified in the configuration. Note that if the server is not listening at the standard port, the correct port value must be configured in **ProxyServerPort** (and **RegistrarServerPort** as needed).

The main purpose of registration is to create and maintain a dynamic binding of the SIP/SP account to the device's local contact address. Service provider can also rely on this periodic message to infer if the device is online and functional. Each OBi device takes only one local IP address that is either statically assigned in the device's configuration, or dynamically obtained from a local DHCP server, or through PPPoE. The method to use to get an IP address assigned is determined by the value of ***WAN Settings*::AddressingType**. The SP*n* services for *n* = 1 – 6 on the other hand each uses a different local contact port for sending and receiving SIP messages (default is 5060, 5061, 5062, …, and 5065 respectively). This port can be configured in the ***SPn Service*::X_UserAgentPort** parameter. **ProxyServer** and **RegistrarServer** must use the same transport protocol for SIP messages that can be set in the **ProxyServerTransport** parameter. The OBi1000 supports UDP, TCP, and TLS for SIP transport. The default server port is 5060 for UDP/TCP and 5061 for TLS. When TCP/TLS is used, the OBi1000 will initiate a TCP/TLS connection only with the **ProxyServer**; all subsequent SIP message exchange between the phone and the servers MUST use the same connection. If for any reason the connection is closed, the phone will attempt to re-establish the connection following an exponential backoff retry pattern.

Note that dynamic address binding through periodic registration is not strictly necessary if the local IP address of the device does not change; the device's contact address may be statically configured on the Registration Server.

Here is a typical REGISTER request generated by the phone:

```
REGISTER sip:as.xyz.broadworks.net:5060 SIP/2.0
Call-ID: 7107d244@192.168.15.207
Content-Length: 0
CSeq: 10722 REGISTER
From: <sip:3134445567@as.xyz.broadworks.net>;tag=SP337b73f3bf7a504c3
Max-Forwards: 70
To: <sip:2404982564@as.xyz.broadworks.net>
Via: SIP/2.0/UDP 192.168.15.207:5062;branch=z9hG4bK-f9e9e56c;rport
User-Agent: OBIHAI/OBi1062-5.0.0.1542
Contact: <sip:2404982564@192.168.15.207:5062>;expires=60;
  +sip.instance="<urn:uuid:00000000-0000-0000-0000-9abcde700065>"
Allow: ACK,BYE,CANCEL,INFO,INVITE,NOTIFY,OPTIONS,PRACK,REFER,UPDATE
Supported: replaces, eventlist, record-aware
```

Note that in the last example, the OBi1000 does not use the Expires header in REGISTER requests. Instead the Expires value (in seconds) is encoded as a parameter in the Contact header. The two methods are equivalent in this usage per RFC3261. Note also that the OBi1000 includes +sip-instance parameter in the Contact header that specifies the phone MAC address in the UUID. This parameter can be suppressed by disabling the option ***ITSP Profile X – SIP***::**X_RegisterIncludeInstance**.

In some cases the device may not receive any response to its REGISTER from the server that is caused by an upstream router blocking the outgoing message sent by the phone. To cope with such case, the administrator may instruct the phone so try other SIP user agent ports by spectify a comma separated list of up to 10 altenative ports in the parameter the ***SPn Service***::**X_UserAgentPorts** and the phone will cycle through those ports to retry REGISTER until a response is received from the server.

Third Party Registration

It is possible to have the phone register for an Address of Record (AOR) that is not the same as the account user-id. That is, the user-id in the TO header of the SIP REGISTER request is different from that in the FROM header, which always carries the account user-id. This is known as third party registration. One application is in the implementation of a shared line using the BLA (Bridged Line Appearance) method. To enable third party registration, set the user-id to register for in the parameter ***SPx Service – SIP Credentials***::**X_ShareLineUserID**.

Registration Period

The *nominal* registration Expires header value (implemented as a Contact header parameter value) used by the phone in REGISTER requests is configured in the parameter ***ITSP Profile X – SIP***::**RegistrationPeriod** (in seconds). The actual expires value is determined by the server. The server may reject the REGISTER request with 423 with a Min-Expires header value (in seconds). The phone will then retry quickly with an Expires header value equal to the Min-Expires header value from the server. When the server accepts the registration, it replies with a 2xx response for the REGISTER and includes an expires parameter value in the Contact header that matches the Contact the phone uses in the REGISTER request. However, if it is not found in the Contact, the phone takes the server supplied expires value from the Expires header of the 2xx response. If still not found, the phone assume the server supplied value is 3600 seconds.

If the server supplied expires value is less than the Expires header value used by the phone, the phone takes the server's version to compute the next renewal interval. Otherwise the phone uses its own Expires header value to do the same. Note that the server should not supply explicitly or implicitly an expires value that is larger than what the phone has asked for, as that would be a protocol violation. The phone however will ignore such error.

The phone computes the next renewal time by subtracting a percentage of the expires value derived from the 2xx response returned by the server. How the margin to subtract is computed can be controlled by the parameter ***ITSP Profile – SIP***::**X_RegistrationMargin** parameter, as follows: If **X_RegistrationMargin** is 0 or not specified, the renewal time is half-way before the expiration, if the expires value is less than 1200s, or 600s before the expiration otherwise. If **X_RegistrationMargin** ≥ 1, it is interpreted as the number of seconds (with any fractional part dropped) before the expiration time to renew registration. If **X_RegistrationMargin** < 1, it is interpreted as the fraction of the current expires value to subtract from the expiration time to get the time for the next renewal. For example, if **X_RegistrationMargin** = 0.01 and the expires value is 300, then the next renewal goes out 3s before the expiration time.

REGISTER Final Non-2xx Response Handling

When registration encounters an error, the phone can schedule retries based on the type of error. Each recognizable error type is represented by a 3-digit code. Error codes 300 – 699 are the error response codes returned by the server, while 900-999 are used to indicate other errors. The following 9xx error codes are related to registration:

- 900 = Timeout waiting for a response from the server
- 901 = Cannot resolve the server name or the host is not reachable

For 3xx class responses with a valid Contact header, the phone will follow the given Contact to retry registration quickly. If a valid Contact is not found, or if the number of consecutive redirections has reached 5, the phone considers the 300 response an error and performs standard error handling.

For 401 and 407 responses with a valid Proxy-Authenticate or WWW-Authenticate header, the phone will retry registration quickly and include the properly computed Proxy-Authorization or Authorization header. However, if there is no valid Proxy-Authenticate or WWW-Authenticate header in the error response and if the number of consecutive 401/407 responses received has reached 2, the phone considers the 401/407 a true error and performs standard error handling.

For 423 responses with a valid Min-Expires header value, the phone will retry registration quickly with a new Expires value that conforms to the Min-Expires value from the server. However, if the Min-Expires header is not present in the response or the value is not larger than the current Expires value sent by the phone, the phone considers it an error and performs standard error handling.

For 5xx – 6xx responses with a Retry-After header, OBi1000 will schedule a retry after the specified value.

The standard handling is by waiting for a certain number of seconds before trying to register again. The number of seconds to wait is detrmined by the rules specified in the parameter ***ITSP Profile X – SIP***::**RegisterRetryResponseCodes** on the actual error code. The format of this parameter is the same as a digit map. Let's consider the default value of this parameter:

```
(<40[17]:w120>|<40[34]:w120>|<99[01]:w120-200>|[4-9]xx)
```

Each rule is a subsitution where a certain error code or error code pattern is mapped to the number of seconds to wait. With this example, the phone waits for 120s for 401 and 407 error codes, 120s for 403 and 404 error codes, randomly between 120 and 200s for 990 and 991 error codes, and a fixed default value for all other error codes. The fixed default value is configured in the ***ITSP Profile X – SIP***::**RegisterRetryInterval** parameter. The syntax `w`{*a*}–{*b*} specifies a random range of between {*a*} seconds and {*b*} seconds. *Error codes not covered by these rules will cause the phone not to retry registration after the error*.

SIP Outbound Proxy Server

An **OutboundProxy** server can be configured on the device such that all outbound requests are sent via the outbound proxy server instead of directly to the **ProxyServer** or **RegistrarServer**. If the outbound proxy server is listening at a non-standard port, the correct port value must be specified in the **OutboundProxyPort** parameter. The **OutboundProxy** may use a different transport protocol from the ProxyServer. The transport protocol to use to communicate with the **OutboundProxy** can be set in the **OutboundProxyTransport** parameters. If **OutboundProxyTransport** is TCP/TLS, the phone will initiate a TCP/TLS connection only with the **OutboundProxy**; all subsequent message exchange between the phone and the servers MUST use the same connection. If for any reason the connection is closed, the phone will attempt to re-establish the connection with the **OutboundProxy** following an exponential backoff retry pattern.

Even though the phone only exchanges messages directly with the **OutboundProxy**, the **ProxyServer**, **ProxyServerPort**, and **ProxyServerTransport** parameters are still very much relevant and important since the SIP requests sent by the phone to the server are still formed based on these values, not based on the **OutboundProxy** value. In fact, the **OutboundProxy** value should never appear in the SIP requests generated by the phone (unless the **OutboundProxy** has the same value as the **ProxyServer**).

Some server implementations will include the outbound proxy server in a Record-Route header such that the phone should not respect the locally configured **OutboundProxy** value after the initial INVITE is sent for a new call. This behavior can be achieved by enabling the option ***ITSP Profile X – SIP***::**X_BypassOutboundProxyInCall**. This option however has no effect when the **OutboundProxyTransport** is TCP/TLS as the phone will always use the same connection to send messages to the server.

DNS Lookup of SIP Servers

When sending out SIP requests to the server the device looks up the IP address of the server, using DNS query if the server is specified as a domain name instead of an IP address. If an Outbound Proxy Server is configured, it is used instead of the SIP Proxy Server or SIP Registration Server. The resolution of the server domain name into IP address is performed in the following manner:

- If ***ITSP Profile X – SIP***::**X_DnsSrvAutoPrefix** is enabled, resolve the name as DNS A Record, DNS SRV Record and as DNS SRV record with a service prefix prepended to the name in parallel by sending 3 queries to each DNS server at the same time. The service prefix to prepend to the name depends on the transport protocol being used; for SIP, `_sip._udp.` for UDP, `_sip._tcp.` for TCP and `_sip._tls.` for TLS. If more than one valid result is returned from the queries, the DNS SRV result for the name with prefix has the highest priority, then the DNS SRV result for the name without the prefix, then the DNS A result

- Otherwise, resolve the name as a DNS A record and as a DNS SRV Record in parallel by sending 2 queries to each DNS server. If both queries return a valid result, the DNS SRV result will be taken over the DNS A result

- If no valid results are returned, the phone considers the SIP request failed with the error code 901

If the result from the DNS query is a SRV record, the server port is also taken from that record (the server port value configured on the device is ignored). Otherwise, the server port is taken from the configured value or 5060 will be used if no value is specified. We recommend setting the **ProxyServerPort** to 0 (i.e. the unspecified value) if DNS SRV lookup is intended for the service.

NAT Traversal Considerations

If the phone is located behind NAT with respect to the service provider equipment, it can discover the mapped external address corresponding to its local SIP contact address as seen by the server. This may help in some cases where a gateway router's SIP ALG implementation is causing communication problems between the device and the server. The device can discover the mapped external address in one of the following ways:

- From the "received=" and "rport=" parameters of the VIA header of the REGISTER response sent by the server; these two parameters tell the device its mapped IP address and port number respectively. This method is used if periodic registration is enabled on the device

- From the response to a STUN binding request the device sent to a STUN server. This method is used by enabling **X_KeepAliveEnable** and setting the **X_KeepAliveMsgType** parameter to "stun". In this case, the STUN server is taken from the **X_KeepAliveServer** parameter, if it is specified. Otherwise, the keep-alive messages are sent to the same server where a REGISTER request would be sent to. The latter is the most effective way of using STUN to discover the mapped external contact address

- From the value of the ***ITSP Profile X – SIP***::**X_PublicIPAddress** parameter.

The discovered external IP address and port, if discovered by one of the methods above, will be used by the phone to replace its private address and port when generating SIP requests, if the ***ITSP Profile X – SIP***::**X_DiscoverPublicAddress** option is also enabled. The substitution of private addresses with public addresses applies to the Contact header of any SIP requests and the `c=` line in SDP. If the option **X_UsePublicIPAddressInVia** is also enabled, the Via address is also substituted (however this is usually not necessary).

The phone can also include an empty rport parameter in the Via header of outbound SIP requests if the option ***ITSP Profile X – SIP***::**X_UseRport** option is enabled. This parameter is sometimes needed to prompt the server to insert an

rport parameter value in the response; it should also prompt the server to send the response to the port where the request originated from (i.e. according to the source port of the IP header of the packet). However, as such behavior on the server is considered standard by many, the empty rport parameter has become superfluous in practice.

Keep Alive Messages

In addition to periodic registration with the server, the phone can be instructed to send out periodic keep alive messages on the same network path to keep the NAT pin hole open. For this purpose, it is recommended the keep alive messages are sent to the same proxy server responsible for registration. The keep alive messages may be dropped by the server unprocessed. However if STUN binding request are used as keep alive messages, it is recommended that the server return a valid STUN binding response to each request. The parameters that control the sending of keep alive messages are:

Parameter Group	Parameter	Description
SPn Service	**X_KeepAliveEnable**	Enable sending of keep alive message. If set to `true`, device sends periodic keep-alive messages to the destination specified in **X_KeepAliveServer** and **X_KeepAliveServerPort**, at the interval specified in **X_KeepAliveExpires**. The content of this message is the ascii string `keep-alive\r\n`
SPn Service	**X_KeepAliveExpires**	Keep alive period in seconds
SPn Service	**X_KeepAliveServer**	Hostname or IP address of keep alive server
SPn Service	**X_KeepAliveServerPort**	UDP port of the keep alive server
SPn Service	**X_KeepAliveMsgType**	The type of keep alive messages to send out periodically if keep-alive is enabled. It can be one of the following choices: - keep-alive: The string "keep-alive" - empty: A blank line - stun: A standard STUN binding request; device will use the binding response to form its contact address for REGISTRATION - custom: use the value of **X_CustomeKeepAliveMsg** (note: option not available on OBi100/OBi110) - options: a valid SIP OPTIONS request. No response to this request from the server will trigger a proxy failover if proxy redundancy is eanbled - notify: a valid SIP NOTIFY request. No response to this request frin the server will trigger a proxy failover if proxy redundancy is enabled.
SPn Service	**X_CustomKeepAliveMsg**	Defines the custom message to be used when **X_KeepAliveMsgType** is "custom". The value should have the following format: `mtd=NOTIFY;event={whatever};user={anyone}` Where - NOTIFY may be replaced by any other SIP method, such as PING, - event parameter is optional and is only applicable if method is NOTIFY. If event is not specified, the 'keep-alive' event will be used with NOTIFY - user parameter is optional; if not specieifed, the request-uri will not have a userid, and the TO header field will use the same userid as the FROM header (which is the local account userid). If user is specified, it will be used as the userid in the Request-URI and TO header. SIP messages for keep-alive are sent only once without retransmission; response to the SIP messages are ignored by the OBi.

SIP Proxy Server Redundancy and Dual REGISTRATION

Server Redundancy specifically refers to the OBi device's capability to a) look for a working server to REGISTER with from a list of candidates, and b) switch to another server once the server that it currently registers with becomes unresponsive. As such, DEVICE REGISTRATION MUST BE ENABLED in order to use the server redundancy feature. Other SIP requests, such as INVITE or SUBSCRIBE, are sent to the same server that the device currently registers with.

If the Outbound Proxy Server is provided, server redundancy is applied to the Outbound Proxy Server instead of the REGISTRATION server. Server redundancy behavior is enabled by enabling the parameter ***ITSP Profile X – SIP***::**X_ProxyServerRedundancy** (which is disabled by default).

Another requirement for using the server redundancy feature is that the underlying server must be configured in the device as a domain name instead of an IP address. This allows the OBi to collect a list of candidate servers based on DNS query. The domain name may be looked up as DNS A record or DNS SRV record. For A records, all the IP addresses returned by the DNS server are considered to have the same priority. For SRV records, the hosts returned by the DNS server can be each assigned a different priority.

After a list of candidate servers are obtained, the OBi device will first look for a working server according to the stated priority. A *working server* means one that the device can successfully register with. This is known as the *Primary Server*. Subsequently, the device maintains registration with the primary server the usual way. However, if no working server is found after traversing the entire list, device takes a short break and repeats the search in the same order.

While maintaining registration with the Primary Server, the OBi will continually attempt to fallback to one of the candidate servers that has higher priority than the primary server, if any. The list of candidate servers that the device is trying to fallback on is known as the *primary fallback list*, which may be empty.

In addition, an OBi device can be configured to maintain a secondary registration with a server that has lower or equal priority than the primary server. Secondary registration can be enabled by setting the parameter **X_SecondaryRegistration** to `true`. If **X_ProxyServerRedundancy** is `false`, however, **X_SecondaryRegistration** does not take any effect. If this feature is enabled, as soon as a primary server is found, the OBi will search for a working secondary server in the same manner from the list of candidate servers that are of lower or equal priority than the primary server. Similarly, once a secondary server is found, the OBi forms a *secondary fallback list* to continually attempt to fallback on if the list is not empty.

The interval for checking the primary fallback list and the secondary fallback list are configured in the parameter **X_CheckPrimaryFallbackInterval** and **X_CheckSecondaryFallbackInterval** respectively. These parameters are specified in seconds and the default value is 60 for both.

Notes:

- Secondary server exists implies primary server exists.
- If the secondary server exists, it immediately becomes the primary server when the current primary server fails; the device then starts searching for a new secondary server if the candidate set is not empty.
- The candidate list may change (lengthened, shortened, priority changed, etc.) on every DNS renewal (based on the entry's TTL). Device will rearrange the primary and secondary servers and fallback lists accordingly, whichever is applicable.

If the server redundancy feature is disabled, the device resolves only one IP address from the server's domain name and will not attempt to try other IP addresses if the server is not responding.

SIP Privacy

The OBi device observes inbound caller privacy and decodes the caller's name and number from SIP INVITE requests by checking the FROM, P-Asserted-Identity (PAID for short), and Remote-Party-ID (RPID for short) message headers. All these headers may carry the caller's name and number information.

If PAID is present, the device takes the name and number from it. Otherwise, it takes name and number from RPID if it is present, or from the FROM header otherwise. RPID, if present, will include the privacy setting desired by the caller. The privacy setting may indicate one of the following options:

- *off* = no privacy requested; the OBi will show name and number.
- *full* = full privacy requested; the OBi will hide both name and number.
- *name* = name privacy requested; the OBi will show the number but hide the name.
- *uri* = uri privacy requested; the OBi will show the name but hide the number.

Regardless, if PAID exists or not, the device always takes the privacy setting from the RPID if it is present in the INVITE request. Note that if the resulting caller name is "Anonymous" (case-insensitive), the device treats it as if the caller is requesting full privacy.

For outbound calls, the caller's preferred privacy setting can be stated by the device in a RPID header of the outbound INVITE request. To enable this behavior, the parameter ***ITSP Profile X – SIP***::**X_InsertRemotePartyID** must be set to `true`, which is the default value of this parameter. OBi only supports two outbound caller privacy setting: privacy=off or privacy=full. The RPID header generated by the device carries the same name and number as the FROM header. If outbound caller-ID is blocked, the device sets privacy=full in RPID and also sets the display name in the FROM and RPID headers to "Anonymous" for backward compatibility. The device will not insert PAID in outbound INVITE requests. You can further instruct the phone to use *sip:anonymous@localhost* in the FROM header by enabling the option **X_UseAnonymousFROM** (that is, the phone will use in this case ***From: "Anonymous" <sip:anonymous@localhost>***).

The phone will also include a ***Privacy: id*** header if **X_InsertPrivacyHdr** is also enabled.

STUN and ICE

The OBi supports standard STUN based on RFC3489 and RFC5389 for passing inbound RTP packets to the device when behind NAT. The parameters that control the STUN feature can be found under the section ***ITSP Profile X – General***::

- **STUNEnable** – Enable this feature (default is `false`).
- **STUNServer** – The IP address or domain name of the external STUN server to use. STUN feature will be disabled if this value is blank, which is the default.
- **X_STUNServerPort** – The STUN Server's listening UDP port. Default value 3478 (standard STUN port).

It should be noted that the STUN feature used in this context is only for RTP packets, not SIP signaling packets (which typically do not require STUN). The device sends out a STUN binding request right before making or answering a call on SPx. If the request is successful, the device decodes the mapped external address and port from the binding response and uses them in the m= and c= lines of its SDP offer or answer sent to the peer device. If the request fails, such as STUN server not found or not responding, the call will go on without using an external address in the SDP.

Standard RTP requires the use of even numbered ports in the m= line. If the external port is not an even number, the device changes the local RTP port, retries STUN and will continue to do this up to 4 times or until a even external port number is found. If the 4th trial still results in an odd external port number, the call will go on without using external address in the SDP.

The OBi supports standard ICE based on RFC5245. ICE is done on a per call basis for automatically discovering which peer address is the best route for sending RTP packets. To enable ICE on the device, set the parameter: ***ITSP Profile X – General***::**X_ICEEnable** to YES (or TRUE). The default is NO (or FALSE).

Note that ICE is more effective if STUN is also enabled. However STUN not a requirement for using ICE on the device. If STUN is enabled and an external RTP address different from its local address is discovered, the OBi offers two ICE candidates in its SDP:

- The local (host) address (highest priority)
- The external (*srflx* or server reflexive) address

Otherwise only the local host candidate is shown in the device's SDP. Note that the device uses the srflx address in the m= and c= lines of the SDP if STUN is enabled and successful.

If ICE is enabled and the peer's SDP has more than one candidate, device sends STUN requests to each peer candidate from its local RTP port. As soon as it receives a response from the highest priority candidate, the device concludes ICE and uses this candidate to communicate with the peer subsequently. Otherwise the OBi allows up to 5s to wait for a response from all candidates and selects the highest priority one that responds. Once ICE is completed successfully, the device will further apply the symmetric RTP concept to determine the peer's RTP address (i.e. send to the address where the peer's RTP packets are coming from).

ITSP Driven Distinctive Ringing

The OBi device offers 10 ring and 10 call-waiting tone patterns in each ring profile. These patterns are numbered from 1 to 10. Each pattern also comes with a configurable name. A different default ring may be assigned to each trunk on the device.

An ITSP can instruct the OBi device which ring to use (by name) for a call routed to SP*n* by inserting an Alert-Info header in the SIP INVITE sent to the device. The Alert-Info must include a URI. For example:

Alert-Info: http://www.xyz.com/some-folder/bellcore-dr4

When the device receives this, it will look for a ring tone name or call-waiting tone name in the ring profile that matches the Alert-Info URI. Ring tone names are not case sensitive when compared. If a match is found, the device plays the corresponding ring or call-waiting tone. Otherwise, the device plays the default ring.

RTP Statistics – the X-RTP-Stat Header

When ending an established call, the OBi device can include a summary of the RTP statistics collected during the call in the SIP BYE request or the 200 response to the SIP BYE request sent by the peer device. The summary is carried in an X-RTP-Stat header in the form of a comma-separated list of fields. The reported fields are:

PS=[Number of Packets Sent]

PR=[Number of Packets Received]

OS=[Number of bytes sent]

OR=[Number of bytes received]

PL=[Number of packets lost]

JI=[Jitter in milliseconds]

LA=[Decode latency or jitter buffer size in milliseconds]

DU=[Call duration in seconds]

EN=[Last Encoder Used]

DE=[Last Decoder Used]

For example:

X-RTP-Stat:PS=1234,OS=34560,PR=1236,OR=24720,JI=1,DU=1230,PL=0,EN=G711U, DE=G711U

To enable the X-RTP-Stat feature, the parameter ***ITSP Profile X – SIP***::**X_InsertRTPStats** must be set to `true`.

RTCP

The OBi1000 supports RTCP (RFC 3550) and RTCP-XR (RFC3611) (with MOS statistics for VQ reporting).

Media Loopback Service

The OBi supports the media loopback draft as described in *draft-mmusic-media-loopback-13.txt*. The following media loopback features are supported by the OBi device:

- Loopback modes: loopback-source and loopback-mirror
- Loopback types: rtp-media-loopback and rtp-packet-loopback
- Loopback packet formats:: encaprtp, loopbkprimer

When acting as a loopback mirror, the OBi device always sends primer packets so that incoming packets can get through NAT or a Firewall. The media loopback feature is controlled by the following parameters (under Phone Settings – Calling Features section):

- **AcceptMediaLoopback** – Enable device to accept incoming call that requests media loopback. Default is YES.

- **MediaLoopbackAnswerDelay** – The delay in milliseconds before the OBi answers a media loopback call. Default is 0.
- **MediaLoopbackMaxDuration** – The maximum duration to allow for an incoming media loopback call. Default is 0, which means the duration is unlimited.

Note that the device will reject an incoming media loopback call if:

- Phone is off hook.
- Phone port is ringing.
- One or more calls are on hold.

The device will terminate an inbound media loopback call already in progress when:

- Phone is off-hook.
- Phone port is ringing.

To make an outgoing loopback call, the user can dial one of the following star codes before dialing the target number:

- *03 – Make a Media Loopback Call.
- *04 – Make a RTP Packet Loopback Call.

Note that outbound Media Loopback Call is not subjected to call duration limit; it will last until the user hangs up or until the called device ends the call.

A SIP/SP Configuration Example

The following table details a configuration example where the ITSP Profile to use is B with two SP services, SP1 and SP3, both pointing to the same ITSP Profile.

Parameter Group	Parameter	Example Value	Notes
ITSP Profile B – General	**SignalingProtocol**	SIP	Standard default value is SIP
ITSP Profile B – SIP	**ProxyServer**	sip.phonepower.com	
ITSP Profile B – SIP	**OutboundProxy**	sbc.phonepower.com	
ITSP Profile B – SIP	**OutboundProxy**		
SP1 Service	**X_ServProvProfile**	B	
SP1 Service – Credentials	**AuthUserName**	3409991003	
SP1 Service – Credentials	**AuthPassword**	i9cik#dkL	
SP1 Service – Credentials	**X_MyExtension**	1003	
SP3 Service	**X_ServProvProfile**	B	
SP3 Service – Credentials	**AuthUserName**	3409991005	
SP3 Service – Credentials	**AuthPassword**	c98dfa74{	
SP3 Service – Credentials	**X_MyExtension**	1005	
Phone Settings	**PrimaryLine**	SP3	

Google Voice™ Service

A Google Voice service is an SP service with Google configured as the "ITSP". The service configuration details are hardcoded internally and there is no need for additional configuration. Parameters such as **ProxyServer** and **OutboundProxy** do not apply when connecting to the Google service.

Google Voice can only be configured via the consumer-facing version of the OBiTALK portal. It cannot be configured locally on the device. To configure the service, log into the OBiTALK portal, add the device and follow the instructions within the portal. All SP services can be enabled for Google Voice, with a different account on each service.

Once Google Voice is enabled on an SP service, the service is bound to an ITSP profile with the parameter ***ITSP Profile X – General***::**Protocol** set to `Google Voice`

Google Voice offers a call screening feature such that you must press digit 1 before answering an incoming GV call. OBi device can be setup to automatically do that for you when you pick up the phone. To enable this feature on the device, set the ***SPn Service – CallingFeatures***::**X_SkipCallScreening** parameter to `true` (`false` is the default).

Please note that the codec is limited to G711u only for all calls.

When Google Voice is selected as the protocol, all the other ITSP Profile parameters are ignored except the DigitMap parameter. The following SP*n* Service parameters are ignored:

- X_Codec_Profile, X_RegisterEnable, X_UserAgentPort, X_UserAgentPorts, X_SipDebugOption, X_SipDebugExclusion
- X_KeepAliveEnable, X_KeepAliveExpires, X_KeepAliveServer, X_KeepAliveServerPort, X_KeepAliveMsgType
- URI, MaxSessions, X_AcceptDialogSubscription, X_AcceptLinePortStatusSubscription

The following features are supported:

- Non-Gmail domain in account name for Google Voice Communications Service.
- Accept DTMF input from a Google Talk client entered by the user as text messages (only 0 – 9, *, and # will be recognized by the device).
- Accept the setting of the parameter ***ITSP Profile X – General***::**DTMFMethod**. The value can be either `InBand` or `RFC2833`. Other values will be reverted to `RFC2833`. Default is `RFC2833`.
- Voice Service Features of the OBi Device.

OBiTALK Service

OBiTALK is a proprietary protocol developed by Obihai Technology for communications among OBi devices and to OBiTALK device management servers. The protocol is intended for two main purposes: a) Peer-to-peer calling between OBi devices, and b) Device management by OBiTALK servers. Every OBi device comes with one instance of the OBiTALK service with the (fixed) factory-assigned 9-digit device OBi Number as the userid of the service. OBi devices can call each other by dialing the other party's OBi Number.

The OBiTALK service is enabled with the parameter ***OBiTALK Service***::**Enable** and is enabled by default (unless it is disabled through ZT Customization).

Through OBiTALK, the user can start calling another OBi device out-of-the-box without any configuration. However, the phone must be able to reach out to the Internet to make an OBiTALK call. To call another number, from the phone dial **9 followed by the 9-digit OBi number. For your convenience, Obihai maintains an *Echo Server* at the reserved OBi Number **222 222 222**. Users can call this number to do a quick *Echo Test* after listening to a short announcement at the beginning. If the test is successful, the user will hear their voice echoed back as they talk during the test. This serves as a rudimentary check that the network and equipment is set up and functioning correctly.

An administrator such as an ITSP may desire to limit OBiTALK calls just to the Obihai Echo Server. To do this the administrator can change the value of ***OBiTALK Service***::**DigitMap** to: `(<ob>222222222|ob222222222)`. Obviously, you can change or add more OBi numbers to this digit map by following the same pattern. A simple way to disable OBiTALK voice calls completely is by setting ***OBiTALK Service – Calling Features***::**MaxSessiions** to `0`; you will not be able to do Echo Test in that case.

The OBiTALK service also makes it possible to view and change the settings of your OBi devices from the OBiTALK portal. If the OBiTALK service is disabled in the device configuration, both OBiTALK voice calls and device management features will not be available.

OBiBluetooth Service

The OBi1000 supports Bluetooth connectivity with either a headset or a mobile phone. While the OBi1062 has built-in hardware support for this function, the OBi1032 requires the user to connect an OBiBT USB dongle to USB Port 2 (note: OBiBT Must NOT be connected to USB Port 1). Other than that, the Bluetooth enabled features work similarly in both models.

To pair the OBi1000 with a headset, the parameter ***OBiBluetooth***::**AudioGateway** must be enabled. When connected with a Bluetooth headset, the user can use the headset for calls just like a wired headset connected to the phone. OBi1000 supports Headset Profile (HSP) Audio Gateway (AG) in this mode. To pair OBi1000 with a mobile phone, on the other hand, ***OBiBluetooth***::**AudioGateway** must be disabled. OBi1000 supports Hands-Free Profile (HFP) Hands-Free Unit (HF) in this mode.

From a user perspective, pairing their device is done via phone screen from the Bluetooth menu item within the Settings/Preferences app. The **Pairing Mode** option gives the choices *Pair with Headset* and *Pair with Mobile Phone*. More information is available in the User Guide for the OBi1000 series.

To use OBiBluetooth Service on the phone, the pairing mode must be set to Pair with Mobile Phone (same as enabling **AudioGateway**) (otherwise the phone considers the service disabled). With that the Bluetooth-connected-mobile-phone becomes a mobile service gateway for the phone to access the mobile service on the mobile phone. Like other services, incoming call to the mobile phone is handled by the phone according to rules configured in ***OBiBluetooth***::**InboundCallRoute**. By default, it is set up to ring just the phone, but you can have it ring the AA or other numbers instead, or in addition.

The administrator may allocate a feature key configured with the Line Monitor function and bound to the OBiBluetooth service, to monitor the status of the OBiBluetooth service. The LED color is solid orange if the mobile phone is not connected, solid green if it is connected and idle, slow blinking green if it is on a call, and fast blinking red if it is ringing. Incoming calls may be answered from the mobile phone or from the OBi1000. If it is answered on the mobile phone, the OBi stops ringing; pressing the flashing Line Monitor key will “move” the call from the mobile phone to the OBi1000. If the call is on the OBi1000, then pressing the soft key To Mobile will move the call back to the mobile phone. Similar operations apply to outgoing calls over the mobile phone whether the call is initiated from the phone or from the OBi1000. Similar operations to move the call between the mobile phone and the OBi1000 may also be initiated from most mobile phones.

The administrator may also allocate a Call Key to bind to the OBiBluetooth service.

Call Features

Phone Level and Line Level Feature

A call feature may be classified as a *Phone Level Feature* or a *Line Level* Feature, or simply referred to as a Phone and Line Feature respectively. A phone feature applies to all calls on the phone regardless which line a call is on. Call Waiting is an example of a phone feature. A line feature on the other hand applies only to calls on the specific line. BLF is an example of a line feature. Some features may have a version for use as a phone feature as well as a line feature that can be used on individual lines. For example, Do Not Disturb can be implemented as a phone feature for all calls and also as a line feature for each of the SP services (SP1-SP6), OBiTALK and OBiBluetooth services.

Call States

As a call progresses from beginning to end, it goes through a number of defined stages commonly known as Call States or States. The following call states are defined for calls on an OBi1000 phone:

Call State	Description	Available Operations	Icon	LED Pattern
Dialtone	Dial tone played to prompt the user to enter the target number to call	End		Steady green
Dialing	The user is entering a target number to call	End		Steady green
Trying	Trying to call the dialed number but the called party has not started ringing yet	End		Steady green
Peer Ringing	The called party is ringing	End		Fast blinking green
Ringing	An incoming call is ringing the phone	Reject, Answer		Fast blinking red
Connected	Connected on a call and both sides are talking	End, Hold, Transfer, Blind Transfer		Steady green
Connected-S	Connected securely (using SRTP) on a call and both sides are talking	End, Hold, Transfer, Blind Transfer		Steady green
Connected-HD	Same as Connected when a wide-band audio codec is in use	End, Hold, Transfer, Blind Transfer		Steady green
Connected-HDS	Same as Connected-S when a wide-band audio codec is in use	End, Hold, Transfer, Blind Transfer		Steady green
Holding	User has placed the call on hold	End, Resume, Add to Conf., Transfer, Blind Transfer		Slow blinking red
Ended	Call failed due to various reasons, such as invalid number, service not available, called party busy, etc.	End (i.e. Remove the Call)		Slow blinking green

Idle	No Call			Off

There are many operations that may be applied to a call during its course. For example, holding, resuming, or ending a call are commonly used operations. The options to manipulate a call are typically presented to the user as Soft Key options, while frequently used call operations can also be mapped to a feature key (such as the Hold and Transfer function). Soft Key options, in particular, are call state sensitive. That is, only the options that are applicable to the call at its current state are shown - as the call transitions from one state to another, the Soft Key options on the screen will be updated accordingly. This will be discussed further in the *Calls App* section.

As stated earlier, each call on the phone must be assigned to a Call Key and each Call Key has its VLKW on the GUI. To help the user identify the current state of a call, each call state is represented by an icon displayed in the respective VLKW. Furthermore, the LED of each call key stays steady or blinks with a certain pattern, with respect to the current call state.

Core Call Features

Line Capacity

Line capacity refers to the maximum number of simultaneous calls that can be active per line; some of the calls may be in the Holding state, except for OBiBluetooth that can only have one call. The configuration of each voice service has a parameter **MaxSessions** that controls how many simultaneous calls to allow on that service. The default value is 2 for all lines. The number should be set to equal to or less than what the underlying service provider can support.

Complex Operations Between Multiple, Diverse Voice Services

The OBi1000 supports call transfer, conference, and call forward operations involving calls on multiple, diverse services. This powerful feature makes the phone very user-friendly when disparate services using different underying technologies are consolidated on the same OBi phone. For example, a user may transfer an OBiBluetooth caller to a Google Voice caller, or call forward from a caller on BroadSoft to a caller on OBiTALK. This will be explained further where we describe the respective call features.

Making Outgoing Calls

Digit Map

A digit map is a succinct way of describing a set of number patterns. The ***Phone Settings***::**DigitMap** parameter determines the set of number patterns that can be dialed by the user. You can refer to named digit maps with the `(M`*`name`*`)` syntax. For example, the default ***Phone Settings***::**DigitMap** refers to digit maps defined for SP1, SP2, SP3, SP4, OBiTALK, and OBiBluetooth services, with the reserved name `(Msp1)`, `(Msp2)`, `(Msp3)`, `(Msp4)`, `(Mpp)`, and `(Mbt)` respectively:

```
([1-9]x?*(Mpli)|[1-9]S9|[1-9][0-9]S9|***|**0|
**8(Mbt)|**1(Msp1)|**2(Msp2)|**3(Msp3)|**4(Msp4)|**9(Mpp)|(Mpli))
```

This way the digit map is more readable and is much better organized. Note that `(Mpli)` refers to the digit map of the Primary Line, which is described later in this document.

Audio Path and On/Off-Hook States

There are three audio paths for calls on the phone: Handset, Speakerphone, and Headset. The headset audio path further supports 3 devices: RJ9 Headset, 3.5mm Headset, and Bluetooth Headset. Only one audio path can be switched

on at a time, and only one headset device can be active when the headset audio path is turned on. The phone is said to be *On-Hook* if none of the audio paths for calls are turned on. Otherwise the phone is said to be *Off-Hook*.

When an audio path is switched on, the other currently active audio path is automatically switched off. The speakerphone and headset audio paths are each enabled by pressing the speakerphone key or the headset key. Lifting the handset from the cradle enables the handset audio path.

Off-Hooking the phone refers to one of the following actions when the phone audio paths for calls are all switched off:

- Lifting the handset from the cradle
- Turning on the speakerphone path by pressing the speakerphone button
- Turning on the headset path by pressing the headset button

On-Hooking the phone refers to one of the following actions when the phone audio is on:

- Placing the handset on the cradle when the currently active audio path is the handset
- Turning on the speakerphone by pressing the speakerphone button when the currently active audio path is the speakerphone
- Turning on the headset by pressing the headset button when the currently active audio path is the headset

Off-Hook Dialing

In this case, the user off-hooks the phone before dialing the number. The phone plays the dial tone when it is off-hook and starts collecting any digits entered by the user and processes these digits according to the phone digit map.

On-Hook Dialing

In this case, user dials the number without off-hooking the phone first (this is also known as pre-dialing). User must press the **Dial** soft-key option to complete the dialing of the number to call. The Phone then processes the entered number according to the digit map.

Outbound Call Routes

After a complete number is collected from the user, the phone determines which service to use for the call by applying the call routing rules defined in the ***Phone Settings***::**OutboundCallRoute** parameter. This parameter may refer to named digit maps defined elsewhere in the phone configuration. The default value as shown below refers to `(Msp1)`, `(Msp2)`, `(Msp3)`, `(Msp4)`, `(Mpp)`, and `(Mbt)` which are digit maps defined for SP1, SP2, SP3, SP4, OBiTALK, and OBiBluetooth services, respectively:

```
{([1-9]x?*(Mpli)):pp},{**0:aa},{***:aa2},{(<**1:>(Msp1)):sp1},
{(<**2:>(Msp2)):sp2},{(<**3:>(Msp3)):sp3},{(<**4:>(Msp4)):sp4},
{(<**8:>(Mbt)):bt},{(<**9:>(Mpp)):pp},{(Mpli):pli}
```

Note that `pli` refers to the Primary Line that is described in the next section.

Primary Line

The Primary Line is the preferred line to use for outgoing calls when the number entered by the user does not include a line-section-prefix (such as **1 or **2). The Primary Line is defined in the parameter ***Phone Settings***::**PrimaryLine** which must be one of the following values (case-sensitive):

- SP1 Service
- SP2 Service
- SP3 Service

- SP4 Service
- SP5 Service
- SP6 Service
- OBiTALK Service
- OBiBluetooth
- Trunk Group 1
- Trunk Group 2

The reserved name `pli` found in the phone **DigitMap** and **OurboundCallRoute** parameters are substituted by the corresponding name of the primary line: sp1, sp2, sp3, sp4, sp5, sp6, pp, bt, tg1, and tg2 for SP1, SP2, SP3, SP4, SP5, SP6, OBiTALK, OBiBluetooth, Trunk Group 1, and Trunk Group 2, respectively.

Explicitly Selecting a Line to make call

If the user does not select a line to use, the phone picks the configured primary line as the preferred line to make the call. The user may explicitly select a line to use in one of many ways:

- Press the corresponding soft-key option hidden under the Lines soft key on the phone screen. There each Line is represented with the number (userid, DID number, or extension) of the Line.
- Press a Call Appearance Key that is bound to the Line (if one is defined and available. See Call Key section)
- Press the Line Monitor Key for the Line (if one is defined)

If the explicitly selected line is different from the configured Primary Line, the phone temporarily reassign the selected line as the primary line when performing digit map validation and call routing for the current call.

Dialing "Speed Dials 99" Numbers

Up to 99 speed dial numbers can be defined in the configuration and the feature is referred to as **Speed Dials 99**. They are dialed as 1 – 99, the corresponding one/two-digit number (i.e. 1 – 99) can be defined in ***User Settings*** - ***Speed Dials***. These 99 speed dial numbers are unrelated to the speed dial feature keys. These numbers can be associated with phones or devices reachable via an Internet or landline service or the OBiTALK network. Be careful with the Speed Dial Set-Up as this may conflict with any Speed Dials that have been set-up on the OBiTALK portal. The Speed Dials that are set-up on the OBiTALK portal will always overwrite anything set-up via the phone connected to the OBi. Note that it is also possible for a user to setup a Speed Dials 99 number using star code (by default *74, *75 and *76 are defined for this purpose).

Dialing Star Codes

Star codes are described in detail in the Star Codes section. Here we only describe how star code is dialed.

Star codes that are interpreted locally by the phone are defined in a star code profile. There are two star code profiles in the configuration: Star Code Profile A and Star Code Profile B, each accepts up to 40 star code definitions. Only one profile can be used and is defined in the parameter: ***Phone Settings***::**StarCodeProfile**.

Called Party Caller ID Display

When making an outgoing call, the phone will attempt to match the called number against the phone book or buddy list to find a corresponding entry. If one is found, it uses that entry to provide a name and picture of the called party to show on the sreen.

Note: If the call is made over a SIP service, it is also possible for the service provider to offer a similar feature by sending called party ID information in a 1xx response to the INVITE or in an UPDATE request sent to the phone before the call is answered. The phone then updates the name and number display accordingly on the screen. This feature is also referred to Calling Line ID.

Handling Incoming Calls

Inbound Call Routes

Incoming calls come to the phone via any configured Line, how the phone should route the incoming call is based on the rules defined in the **InboundCallRoute** parameter of each Line. The typical way to route an incoming call is to ring the phone; so the default **InboundCallRoute** value for all services is, simply, `ph`.

Rejecting Incoming Calls

To reject a ringing call, press the "Reject" soft key as prompted by the phone display.

Caller ID Display

Caller's name and number are displayed on the screen when available. When only the caller number is available, the phone will attempt to match the number against the phone book or buddy list to find a corresponding entry. If one is found, it uses the name and picture from that entry to display on the screen.

Note: If the call comes from a SIP service, the service provider may provide caller-id picture in the INVITE request by including a Call-Info header with a (http or https) URL to the picture file and the attribute "purpose=icon". For example:

Call-Info: <http://abc.com/user/somepic.png>**;purpose=icon**;**org="**ABC Publishing**"**

These are the acceptable picture formats: png, bmp, gif, jpg. Notice that in the last example, we also include an org attribute in the same header. The value is used to propagate the Organization field in the Caller-ID display.

With another Call-Info header, you can also specify an Action URL to show further information about this caller when the user presses the **ciurl** Soft Key (you must include this key in one of the applicable Soft Key Set):

Call-Info: <https://abc.com/user/info.php?user=john.j.smith>**;purpose=info**

Ending Calls

For calls in any state, user can end the call by pressing the **End** soft key. For calls in connected state, user can end the call simply by hanging up (i.e. on-hooking the phone).

Holding Calls

While a call is connected, the user may place the call on hold using one of the following methods:

- Pressing the Hold soft key; this holds only the highlighted call on the screen
- Pressing the feature key that has been assigned the Hold function; this holds all calls that are in a hold-able state

Resuming Calls

While a call is holding, the user may resume the call using one of the following methods:

- Pressing the "Resume" soft key; this resumes only the highlighted call on the screen
- Pressing the call key that hosts the call to resume; this resumes the call only if it is in the Holding State as indicated by the key's LED blinking pattern (slowly in red)

Note that unlike Hold, there is no Resume function that can be assigned to a feature key. Instead a "conference" function is available for resuming calls to join a conversation, which is discussed when we describe the multi-party conference call feature.

"Foregrounding" a Call

Operations that are said to foreground a call (i.e., bring a call to the "foreground") include:

- Resuming a holding call using the **Resume** soft key

- Answering a ringing call
- Initiating off-hook dialing of a new call
- Starting a new call by on-hook dialing or calling from speed dial, phonebook, or call history, etc.

Right before carrying out a foregrounding operation, the phone automatically applies the following “backgrounding” operations on all other calls in the system based on their respective call states at that moment:

- Calls in Connected State: hold them
- Calls in Ringing or Holding State: leave them alone
- Calls in other states: end them

Call Waiting

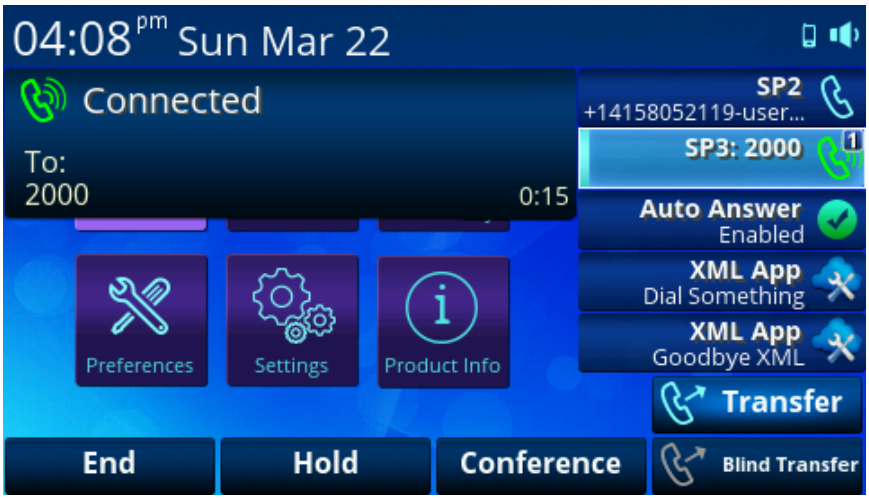
Call Waiting refers to when there are one ore more new incoming calls while there are one or more calls in the other states. If call waiting is disabled, all the new incoming calls are rejected as busy. If call waiting is enabled, incoming calls will ring the phone if there are no connected calls. Otherwise the phone will play the call waiting tone.

Call Transfer

Transferring a current call with remote party A on the phone involves 1) calling another remote party B known as the *transfer target* and 2) connecting A and B while the phone user himself drops out of the conversation. Remote party A is known the *transferee* while the local OBi1000 user is the *transferor*. There are three types of call transfer:

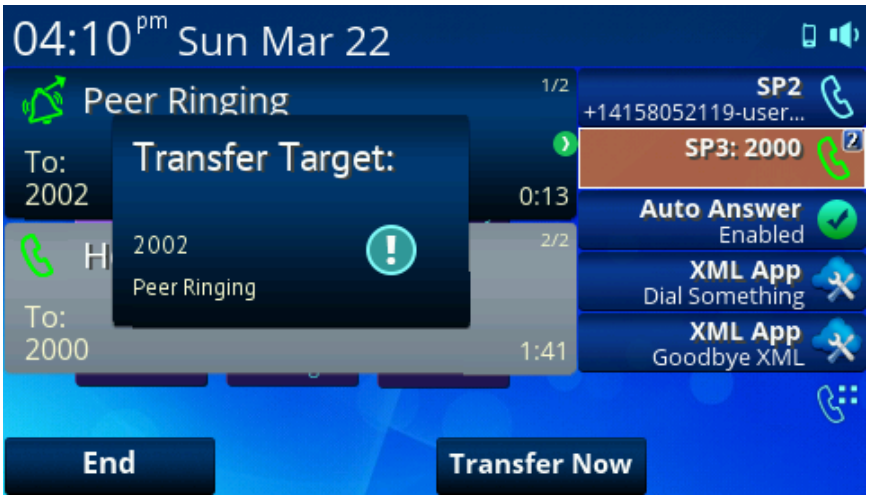
1. **Attended Call Transfer**: In this case, the OBi user waits for B to answer and talks to B first before completing the call transfer. The OBi user presses the "Transfer Now" soft key to complete the call transfer. This is sometimes called consulted transfer or transfer with consultation. Whereas the two types described below are sometimes known as transfer without consultation.
2. **Semi-Attended Call Transfer**: In this case, OBi user waits for B's Phone to ring but before B answers to complete the call transfer. The OBi user presses the "Transfer Now" soft key to complete the call transfer.
3. **Unattended (or Blind) Call Transfer**: In this case, the call transfer is completed by the phone as soon as the OBi user enters B's number, without waiting for it to ring. That is, the OBi user does not need to press any additional key to complete the call transfer. The OBi user would not know if B is busy, or if the number entered is valid or not.

To transfer a call, user must first identify the call to transfer by highlighting the corresponding entry in the call list of the Calls App. Hence the Calls App must be running on the top of the screen at the first level. The call must also be in a transferrable state, which is either the Connected or Holding state. If the highlighted call is transferrable, the soft keys include the "Transfer" and "Blind Transfer" options. Select the "Transfer" option to perform an attended or semi-attended call transfer, or select the "Blind Transfer" option to perform a blind call transfer. Below is an example of a connected call with the Transfer and Blind Transfer soft keys shown.



Pressing the Transfer soft key starts off-hook dialing to collect the transfer target number from the user and subsequently makes the call to the dialed number. As such the phone applies standard handling of all current calls per Section 4.7 before starting dial tone and popping up a dialog box to let the user enter the transfer target number.

The OBi user can abort the call transfer operation any time before the call transfer completes, but not after. While entering the target number, the transfer operation is aborted if the user dismisses the Dial Dialog before a complete number is entered. On the other hand, as soon as a complete transfer target number is entered, the transfer operation may not be aborted in the blind transfer case (i.e. the type started by user pressing the "Blind Transfer" soft key). In the attended and semi-attended cases, the phone shows the Complete Call Transfer dialog along with the "Transfer Now" soft key option as shown in the picture below.



The user may abort the pending transfer operation by dismissing this dialog (with the Home or Cancel Key). In that case, the pending transfer operation is canceled, but the call to the transfer target continues, albeit reverted to a regular outgoing call. The user must explicitly end the call to the transfer target if he does not wish to continue with it. Ending the call to the transfer target at any time before call transfer completes on the other hand cancels the pending call transfer. The original call with the transferee which may have been placed on hold automatically as a result of starting a call transfer is not automatically resumed either when call transfer is canceled; the user must explicitly resume the call to reconnect with the the transferee party. Note also that if the transfer operation eventually fails after completion (from the OBi user's perspective), the original call with the transferee cannot be restored.

Transfer Signaling

As transferor, if the call with the transferee is on SPm and the call with the transfer target on SPn where SPm and SPn point to the same ITSP Profile *X* (including the case $m = n$) and ***ITSP Profile X – General***::**SignalingProtocol** is SIP and ***ITSP Profile X – SIP***::**X_UseRefer** is `true`, the phone will attempt to transfer the call by sending a REFER request to the transferee with the transfer target's SIP URL in a Refer-To header. For other cases, the phone will try to bridge the transferee and transfer target call legs internally.

Limitations of Transfer by Internal Bridging

The phone acts as a proxy of RTP packets sent by each peer of the bridge, without any transcoding. While the phone will try its best to negotiate the codec to use with each call peer that would be acceptable by the other peer, the administrator should understand this limitation and configure the codec profiles accordingly. For example, if Google Voice is to be used, then a call transfer involving a Google Voice call leg must make sure the other call leg supports the G711U codec.

Conference Calls

A conference call is a conversation involving 2 or more remote parties. In order to start a conference, there must be at least two calls and with at least one of them in the Connected State and one of them in the Holding State . The following picture shows such scenario with two calls. To start a conference, highlight one of the calls in the Holding State and select the **Add to Conf** soft-key option.

OBi1000 phones support two methods to conference multiple parties: a) local mixing/bridging and b) external conference bridge. The user interface is slightly different in each case as described below.

Local Mixing/Bridging

After starting a 3-way conference as described earlier, the user can see the two remote parties both in the Connected State.

The OBi1000 supports up to 4-way conferencing (with 3 remote parties) using the local mixing/bridging method. To add a third remote party, make a new call to the target party (or answer a new incoming call from another remote party, if applicable), which automatically places the two connected calls on hold. When the target party rings or answers, resume the two original conferees (one at a time by highlighting each holding call on screen and applying the **Add to Conf** soft-key option to it). Eventually you will have a 4-way conference with three calls in the Connected State.

With locally mixed N-way calls, user can see N individual call items on the screen, one for each call-leg of the N-way call. Hence the user can also individually control each call-leg, such as holding or ending any one of them. Two additional features that are useful in this scenario are Call Selective Mute and Coachee Mode as described below.

Call Selective Mute

On a locally-mixed N-way call, user can selectively mute any of the call legs (i.e. no audio sent to the call peer only), using the **Mute** softkey on the highlighted call on the screen. The **Mute** softkey can be enabled for the **CallConnected** and **CallConnecting** soft key sets.

Coachee Mode

On a locally-mixed N-way call, user can select one of the call legs to be her coach using the **Coachee** softkey (i.e., the user herself will become the coachee to the peer on that call leg). In coachee mode, the audio of the coach is NOT mixed into the audio sent to the other call peers of the N-way call; the coach audio is only added to the mix to the local user, the coachee. The audio mix to the Coach on the other hand includes all the other parties. The **Coachee** soft key can be enabled for the **CallConnected** and **CallConnecting** soft key sets. For the cocahee, the call item on the screen can show an icon to indicate if a call-leg is currently in coachee mode. The coach on the other hand has no knowledge of her role as the coach unless the the call is on OBiTALK service. In that case, the coach's phone can also show a Coach icon in the call item to indicate its special role in that call.

External Conference Bridge

(SIP/SP Only.) When using an external conference bridge, the conference size is not limited by the phone but by the conference bridge. Check with your conference service provider on the conference size limit. Again you can start a 3-way conference as described earlier. In this case the phone first sends a new INVITE to the SIP URL of the conference bridge to request the conference resources. If successful, the conference bridge replies a 2xx response with a Contact header that includes the context information for other conferees to access the bridge for this conference call. On that the phone maintains the call with the bridge in the Connected State and sends a REFER to both conferees to refer them to the Contact as referenced by the conference bridge. The phone user will then notice that only one connected call to the conference bridge remains on the screen while the two calls with the initial two conferees are removed. Presumably the two conferees have also connected to the conference bridge on their own.

To add another conferee, make a new call to the target number (or answer a new incoming call from another party, if applicable), which automatically holds the call to the conference bridge. When the called target rings or answers, highlight the call with the conference bridge (currently in the Holding State) and select the **Add to Conf** soft-key option. At that point the call to the conference bridge is resumed while the new remote party is (or will be if it is still ringing) referred by the phone to the same conference bridge Contact (as soon as it answers) to be added to the conference bridge. You can continue to add more conferees this way until it reaches the limit set by the conference bridge.

To enable external conference bridge operation, insert the rule `{cbridge:sp1(conference)}` in ***Phone Settings***::**OutboundCallRoute** and also enable the option ***Phone Settings***::***Calling Features***::**UseExternalConferenceBridge**. It should be noted that the phone assumes that only conferees that are on the same SP service or using the same ITSP profile as the conference bridge can be referred to the bridge. For conferees that are referable, the phone keeps them in the conference using local mixing and will be subject to the local mixing limit. For example if you have a conferee that is connected through the OBiBluetooth service, the phone keeps the call with that conferee in the Connected State as well as the call to the conference bridge in the Connected State and applies local mixing to the two calls.

No Hold Call (NOHC)

This feature allows the user to start a new call while having one or more active calls without placing other calls on hold first. Hence the current call peers will not notice if any state changes (and could continue to talk without any interruption). User starts a *no hold call* by pressing the NOHC softkey, which may be enabled for the **CallConnected** soft key set.

Expanded Call Features

Expanded call features go beyond the basics to provide additional features for enhanced productivity and control. Unless noted otherwise, the features described here can be performed by the phone independent of a soft switch.

Auto Answer and Intercom

(SIP/SP Only.) Intercom, sometimes (called 2-way) paging, refers to a call with the following characteristics:

1. The called phone answers automatically, usually immediately without ringing. If the handset is on the cradle, the speakerphone or the headset will be turned on automatically as the call is answered. Typically the answering phone also plays a short beep to alert the called user right after answering the call. Use the option ***User Preferences***::**IncomingPageAlertTone** to control whether the phone plays a beep when auto-answering. You can also tell the phone auto-mute when auto-answering with the option ***User Preferences***::**AutoAnswerAutoMute**.
2. The calling phone may support PTT (Push-To-Talk) in addition such that the call ends as soon as the caller releases the respective PTT key on the calling phone. PTT does not apply to the called phone on the other hand; the called user talks and ends the call normally just like any other call

We refrain from referring to Intercom calls as paging in this document so that it will not be confused with another paging application called Page Groups that is based on Multicast/RTP/RTCP. We will talk more about Page Groups later in this document.

The phone supports two methods to signal to the called device to auto-answer the call:

- Call-Info: The phone inserts an ***answer-after=0*** parameter in a Call-Info header in the INVITE request
- Alert-Info: The phone inserts ***info=alert-autoanswer;delay=0*** parameters in an Alert-Info header in the INVITE request

The method to use is controlled by the parameter ***ITSP Profile X – SIP***::**X_AutoAnswerMethod**.

Regardless of the method selected, for incoming calls, OBi1000 processes the "answer-after" parameter in a Call-Info header or the "info" and "delay" parameters in an Alert-Info header, whichever is present in the inbound INVITE, by automatically answering the call after ringing for the number of seconds specified in those parameters (usually 0). When the call is automatically answered, all other current calls on the phone will be interrupted the standard way as this is a new call added to the foreground. The auto-answer behavior can be turned off by disabling the option ***Phone Settings – Calling Features***::**AutoAnswerEnable**. The setting can be changed by the user from the phone under the Preferences menu. The administrator may further define a feature key with the function `Auto Answer On/Off` to give the user a shortcut to disable auto-answering intercom calls whenever he does not want to be interrupted. The color of the LED reflects the current auto-answer on/off state as a visual reminder to the user. If **AutoAnswerEnable** is turned off, the intercom call will be presented to the user as a normal incoming call and will ring the phone normally.

Depending on the method selected, for outgoing calls, the OBi1000 may request the called party to auto-answer by inserting either "answer-after=0" parameter in a Call-Info header or "info" and "delay" parameters in an Alert-Info header in the outbound INVITE request. This behavior can be invoked by dialing a star code that invokes the barge-in (or bar for short) action (*96 by default) before the call and it only applies to the next immediate outgoing call. However, the soft-switch may or may not recognize this parameter or pass it through to the called phone.

A soft-switch may have its own special way of letting a phone user invoke auto-answer feature on the called party. For example, FreeSwitch uses the feature code 8+{extension} to signal auto-answer. For example, user can dial 81002 to request extension 1002 to auto-answer the call.

Auto-Answer Incoming Call Based on Inbound Call Routing Rules

You can let the phone auto answer certain incoming calls based on inbound call routing rules by specifying a rule that routes the call to ph(autoans). For example:

{someid:ph(autoans;nobeep)},{(@.4089991234):ph(antoans;delay=2)},{ph}

The autoans syntax supports two optional semi-colon separated attributes:

- nobeep: Not to play a beep tone on answering. Default is to play the beep tone according to user preference setting
- delay={value in seconds}: The number of seconds to ring before auto-answering the call. Default is 0

Auto-Answer when Caller Requests to Barge-In (with OBiTALK Service)

In this scenario, the caller requests the called phone to auto-answer in order to join (or barge-in) an active call currently on the called phone. Many soft switches offer this feature to the user by performing the audio mixing at the server end without the phone knowing anything about the operation.

With proprietary signaling the OBi1000 on the other hand offers the Barge-In feature also at the device end with the OBiTALK service (that is available to OBi-to-OBi calls only). The calling phone will request to barge-in when the user dials a star code to enable the star code variable $barge-in1 before dialing the target OBi number. The following star code setup example uses *98 for this purpose:

```
*98, Barge In, set($barge-in1,1)
```

The called phone will automatically answer and conference in the calling phone, subjecting to resources available at that moment. Note that the barge-in operation is only applicable if there is at least one active call on the called phone; otherwise the incoming call to barge-in will be rejected.

Barge-In as a Coach

The calling phone to barge in may in addition request to coach the called phone instead of fully participate in the active call. The calling peer that barges in is known as the **Coach**, while the called peer is known as the **Coachee**. The original peer at the other end of the active call that the cocahee has been talking to is referred to as the **Patron**. Precisely, coaching means: The coach hears a mix of the cocachee and patron speeches, the coachee hears a mix of the coach and patron speeches, but the the patron hears only the coachee. To request to coach the called peer, the coach dials a star code to enable the star code variable $coach1 before dialing the coachee OBi number. The following star code setup example uses *99 for this purpose:

```
*99, Barge In, set($coach1,1)
```

Note that this feature is also known as 'Whispering" In some literature.

Push To Talk

The phone supports push-to-talk mode with the feature key function Speed Dial, Busy Lamp Field, and Page Group 1 and 2. See the corresponding feature key section on how to enable the PTT mode with each function.

Speed Dial Feature Key

The phone administrator may allocate one or more feature keys on the phone to be used as Speed Dials. To find out which feature keys are set up as Speed Dials, you can:

- For a VLK, check the function icon on the respective VLKW on the display
- For any feature key, press and hold down the key until the respective feature key item is shown on the display. Then check the function icon of the feature key item.

Note that only the administrator can designate the function for each feature key. The user is not able to change the function of a feature key as assigned by the admin. The user can however configure the **Number** and **Service** parameter of a speed dial feature key from the phone. The user can press and hold down the key until the corresponding feature key item is shown on the display and then enter view or change the target number and the service (a.k.a. line) to use when calling from that speed dial key. To call from a speed dial key, simply press and release the feature key normally.

- To enable the PTT mode on a speed dial, the administrator must include the ptt flag in the Number parameter of the speed dial.

The general syntax is **Number** = {*target-number*}[`;ptt`][`;send=`{*digit-codes*}] where {*target-number*} can be an empty value if the number is unassigned, and {*digit-codes*} is a sequence of the following case-sensitve codes:

- 0-9,*,#,a,b,c,d – The DTMF digit to send to the peer. Each digit is sent with 100 ms on and 100 ms off
- S – Pause for 3s

- s – Pause for 1s
- U"{prompt}" – Prompt the user to enter one more digits manually on the phone with the given {prompt} shown on the screen. User then press "OK" soft key to continue
- A – Wait for the called party to answer before continuing

Note that the phones starting executing the first code in {digits} when the call receives early media or when the call is answered, whichever happens first.

For example: `send=Ass1234U"Enter Passcode"5678`

Note that if a voice service is specified for the speed dial feature key, either in the Service field or the value of the Number field is a full number (i.e. one that includes voice service information), the phone does not apply digitmap before calling the speed dial. Hence any *codes in the number are not processed and the result may not be desirable. You can include the *codes to be processed by the phone by enclosing them in a pair of [...] at the beginning of the number field. For example: you may set the number field to `[*96]SP1(2113)`. The phone then intepprets *96 locally (to request auto-answer on the called party) and makes the call to 2113 using SP1 service.

Block Caller ID*

(SIP and OBiTALK only.) Block Caller ID (or Anonymous Call) allows you, when making an outgoing call, to prevent your name and number information from appearing on the called phone. This is a feature to can be enabled for all calls on the phone or only for calls on a specific service. For calls on an SP service, the feature may be offered locally by the phone, or by the soft-switch serving that SP service on the phone. For SP1 – SP6, the parameters that are related to this service are (where the *X* in ITSP Profile *X* is the value of **ServProvProfile** of the line):

Parameter Group	Parameter	Description
ITSP Profile X – SIP	**X_InsertRemotePartyID**	This is a Boolean option. If enabled, OBi includes Remote-Party-ID header in the INVITE. This header contains similar caller information as an unmasked FROM header. In case the FROM header is masked, the user information in this header may be used for authentication or accounting purpose
ITSP Profile X – SIP	**X_InsertPrivacyHdr**	If enabled, OBi includes a Privacy: id header when anonymous call is enabled
ITSP Profile X – SIP	**X_UserAnonymousFrom**	If enabled, the FROM header is totally masked
SPn Service – Calling Features	**AnonymousCallEnable**	Enable Anonymous Call feature on this line
SPn Service – Network Provided Services	**AnonymousCall**	Anonymous Call service is provided by the soft-switch. The GUI sets and gets the setting at the service provider instead.
Phone Settings	**AnonymousCall**	Enabling this option is equivalent to enabling the feature for all services.

*This service requires ITSP support. While most ITSP services support this service, Caller ID Blocking is NOT available with Google Voice service at present.

If the feature is offered by the phone for the SIP/SP service, it does the following to the outgoing INVITE:

From: "Anonymous" <sip:12345@abc.com>;tag=spx13040-5345425

From: "Anonymous" <sip:anonymous@localhost>;tag=spx13040-5345425

Privacy: id

If the feature is offered by the soft-switch, the phone does nothing but to make the option available to view and set by the user from the GUI.

The feature can be made available in several ways by a combination of phone and soft switch, which we will discuss later. When the feature is available, the administrator will let the user enable or disable the feature from the phone using one of the following methods:

- Using a star code: By default *67 to use Caller ID Block for one call only, dial *67 and then the destination number. To use Caller ID Block on a persistent basis, dial *81 from the handset attached to the OBi. All calls will use the Caller ID Block feature until you cancel the Caller ID Block. To cancel Caller ID Block, dial *82 from the handset attached to the OBi
- Using a feature key: Administrator defines a feature key with the function "Block Caller ID"

Block Anonymous Call

Block Anonymous Call allows you to block incoming calls that has have no identifying caller ID number. Incoming calls will be presented with a busy signal or busy call treatment (such as Call Forward On Busy). The administrator can make this feature available to end-user configurations with one of the following methods:

- Star code: To use Anonymous Call Block, from the phone attached to the OBi, dial *77. To cancel Anonymous Call Block, from the phone attached to the OBi, dial *87.
- Feature Key with the function "Block Anonymous Call"

This feature has both Phone version and Line versions.

Calling Line ID Display

(SIP/SP only) When making an outgoing call, OBi1000 accepts SIP UPDATE method from the Soft-Switch that may contain the called party's name and number. OBi1000 then updates the phone screen with the called party information from the SIP UPDATE request. There are no configuration parameters for this feature.

The OBi can also identify the called party ID from the 18x and 2xx responses to the outbound INVITE message.

Call Forwarding

Call Forwarding allows you to send incoming calls to another number of your choosing. Calls can be forwarded to a number reachable from the landline service, VoIP service or OBiTALK network. Three types of call forwarding are supported: Call Forward Unconditional (same as Call Forward All), Call Forward On Busy, and Call Forward On No Answer.

There is one set of Call Forward Settings per Line and one additional set at the phone level. Incoming calls on a particular Line are processed by the call forwarding rules at that line level first, then at the phone level, whichever is applicable.

Call Forward Numbers

Calls may be forwarded to numbers on the same service or on another service. Therefore each call forward number stored in the OBi configuration MUST include call routing information to let the device know which voice service should be used to forward the call to. The general format of a call forward number is:

TK(number)

where *TK* is the abbreviated name of a voice service. Valid values of *TK* are SP*n* for the SP*n* Voice Service where *n* = 1 – 8, BT1, BT2 for OBiBluetooth 1/2, or PP for the OBiTALK Service.

The *number* to forward to must be in the final form that is acceptable by the service provider. The OBi will not apply any Digit Map or Call Routing Rules on it.

Examples: SP1(14089991234), PP1(ob200333456)

Call Forward ALL

When you use Call Forward ALL, all calls are immediately forwarded to the number you indicate when you enable the feature. To enable Call Forward ALL, from a phone attached to the OBi, dial *72. You will be prompted to enter the number to which the calls will be forwarded. Dial the number plus the # key and a confirmation tone will be heard. To disable Call Forward ALL, dial *73. A confirmation tone will be heard.

Call Forward on Busy

When you use Call Forward on Busy, all calls are forwarded to the number you indicate only when you are already engaged in a call. To enable Call Forward on Busy, from a phone attached to the OBi, dial *60. You will be prompted to enter the number to which the calls will be forwarded. Dial the number plus the # key and a confirmation tone will be heard. To disable Call Forward on Busy, dial *61. A confirmation tone will be heard

Call forward on No Answer:

When you use Call Forward on No Answer, all calls are forwarded to the number you indicate only when you do not answer the call. To enable Call Forward on No Answer, dial *62. You will be prompted to enter the number to which the calls will be forwarded. Dial the number plus the # key and a confirmation tone will be heard. To disable Call Forward on No Answer, dial *63. A confirmation tone will be heard.

Parameter Group	Parameter	Description
ITSP Profile X – SIP	**X_Use302ForCallFoward**	
SPn Service – Calling Features	**CallForwardUnconditionalEnable**	Enable Call Forward Unconditional feature on this line
SPn Service – Calling Features	**CallForwardUnconditionalNumber**	Call Forward Unconditional Number. Must be in full number format
SPn Service – Calling Features	**CallForwardOnBusyEnable**	Enable Call Forward On Busy feature on this line
SPn Service – Calling Features	**CallForwardOnBusyNumber**	Call Forward On Busy Number. Must be in full number format
SPn Service – Calling Features	**CallForwardOnNoAnswerEnable**	Enable Call Forward On No Answer feature on this line
SPn Service – Calling Features	**CallForwardOnNoAnswerNumber**	Call Forward On No Answer Number. Must be in full number format
SPn Service – Network Provided Services	**CallForwardOnNoAnswerRingCount**	
SPn Service – Network Provided Services	**CallForwardUnconditional**	
SPn Service – Network Provided Services	**CallForwardOnBusy**	
SPn Service – Network Provided Services	**CallForwardOnNoAnswer**	

Call Forward Signaling

If the called service is on SP*m* and the call forward target is on SP*n* where SP*m* and SP*n* point to the same ITSP Profile *X* (including the case *m* = *n*) and ***ITSP Profile X – General***::**SignalingProtocol** is SIP and ***ITSP Profile X – SIP***::**X_Use302ForCallForward** is `true`, the phone will attempt to forward the caller by replying a 302 response to the INVITE request with the forward target's SIP URL in a Contact header. For other cases, the phone will try to bridge the caller and forward target internally.

Limitations of Call Forward by Internal Bridging

The phone acts as a proxy of RTP packets sent by each peer of the bridge, without any transcoding. While the phone will try its best to negotiate the codec to use with each call peer that would be acceptable by the other peer, the administrator should understand this limitation and configure the codec profiles accordingly. For example, if Google Voice is to be used, then a call forward involving a Google Voice call leg must make sure the other call leg supports G711U codec.

Do Not Disturb

Do Not Disturb (DND) allows you to set the phone to immediately forward calls made to your OBi to the number set-up as your voicemail number / account. If no voicemail account is set-up, the OBi will return a busy signal to the caller until you turn off DND. To turn on DND, from a phone attached to the OBi, dial *78. To turn off DND, from a phone attached to your OBi, dial *79.

Parameter Group	Parameter	Description
SPn Service – Calling Features ***n=1-6***	**DoNotDisturbEnable**	Enable Do Not Disturb feature on this line
SPn Service – Network Provided Features ***n=1-6***	**DoNotDisturb**	
Phone Settings	**DoNotDisturb**	
Phone Settings	**DoNotDisturbFeatureProvider**	

Do Not Ring

Enabling Do Not Ring disables the ringer – this is equivalent to silent mode. The phone screen will still indicate when a call is incoming. A feature key can be assigned with the corresponding LED indicating the Do Not Ring on/off status.

Message Waiting Indication – Visual and Tone Based

(SIP/SP Only.) Message Waiting Indication notifies the user when a new voice message is available. The OBi supports both Visual and Tone based Message Waiting Indication. With Tone-based Message Waiting Indication, the user is alerted to a waiting message by a "stutter" dial playing when the phone is taken off-hook. Typically, this stutter tone will be removed once the user listens to the message(s). Visual-based Message Waiting Indication will turn on a light and show a voicemail icon on the screen when there is a message waiting.

Parameter Group	Parameter	Description
SPn Service – Calling Features ***n=1-6***	**MWIEnable**	Controls the stutter tone
SPn Service – Calling Features	**VMWIEnable**	Controls the Message Waiting Light at the top edge of the OBi1000

n=1-6		
SPn Service – Calling Features *n=1-6*	**MessageWaiting**	This is the message waiting state. It is configurable so that the administration can clear the status if necessary. However, administrator normally should not provision this parameter
ITSP Profile X – SIP	**MWISubscribe**	
ITSP Profile X – SIP	**MWISubscribeExpires**	

Accepts unsolicited NOTIFY for event message-summary

May subscribe to message-summary event package and process NOTIFY within the subscription dialog.

Multicast Page Groups

A paging group is a multicast group that every phone, on the same LAN, that has joined the group can send audio to and receive audio from the group. A multicast group is defined with a multicast address. Each OBi1000 phone supports two multicast groups that are configurable by the admin. The admin may also assign a feature key for users to join or leave each group. The default settings on the OBi work in most LAN environments, so usually all that is required is for the feature to be enabled and a feature key mapped to each phone for the service. Push to talk can also be enabled for Page Groups – this can be configured under ***Phone Settings***::**Page Group 1/2**. Consult your phone administrator if to see if the group paging feature is enabled at your location.

Here is how to use a group paging feature key:

1. If not joined (LED off), press the key once to join. Once you joined, LED is red and your phone will play audio received from the multicast channel
2. After joining (LED red), press and hold down the key to talk (PTT or push-to-talk) to the group. Audio received from the multicast channel is paused while you are talking
3. If clamp-on option is enabled by the admin, you can enable clamp-on so that you can continue to talk without needing to keep pressing down the key. To enable clamp-on, after joining, press and release the key quickly and the LED will change to blink slowing in red to indicate clamp-on enabled
4. To leave the group after joining (and clamp-on, if available), press and release the key quickly

Parameter Group	Parameter	Description
Page Group N *N = 1, 2*	**GroupName**	A user friendly name to label the group on the phone GUI
Page Group N *N = 1, 2*	**MulticastAddress**	Must be a valid IPv4 Multicast Address. Default is `224.1.1.100` for both paging groups
Page Group N *n = 1, 2*	**MulticastPort**	Default is `65322` for page group 1 and `65324` for page group 2. Note that each group must use a different port number
Page Group N *N = 1, 2*	**PushToTalk**	A Boolean option to control whether the page group key (soft key or feature key) should act as a PTT key. That is, user can talk only while pressing+holding the key down. Default is `false`.
Page Group N *N = 1, 2*	**TTL**	TTL value of outgoing (multicast) RTP packets. Default is `2`.
Page Group N *N = 1, 2*	**AudioCodec**	Audio codec to use for outgoing page. Default is `G711U`
Page Group N	**TxPacketSize**	Outgoing RTP packetization in

N = 1, 2		milliseconds. Default is `20`
Page Group N *N = 1, 2*	**RTCPTxInterval**	Interval in milliseonds between sending outgoing RTCP packets when paging. No RTCP packets sent if value is 0, which is the default. Note that a RTCP Bye packet is always sent when ending an outgoing page regardless of this setting.
Page Group N *N = 1, 2*	**SilenceSuppression**	A Boolean option to control if Silence Suppression should be used for outgoing page. Default is `false`.
Page Group N *N = 1, 2*	**ParticipantName**	A string value to identify the current user who is paging to the group in outgoing RTCP packets
Page Group N *N = 1, 2*	**PlayToneOnIncomingPage**	A Boolean option to control whether to play a short "Paging Tone" before playing a new incoming page. Default is `true`.
Page Group N *N = 1, 2*	**PreferredAudioDevice**	Select the preferred audio device to play incoming page. Enter either `System`, `Headset`, or `Speaker`. Default is `System` to let the phone picks according to other user preferences
Page Group N *N = 1, 2*	**PageTimeout**	Limit of outbound page duration in number of seconds to avoid accidentally jamming the page group channel. The duration would be unlimited if the value is 0 which is also the default.
Page Group N *N = 1, 2*	**EnableJoinLeaveGroupUserControl**	A Boolean option to control whether to allow the user the option to Join/Leave a page group by pressing (and releasing it quickly) the pg1/pg2 soft key. Default is true. When it is set to false, press the key once to talk and press it again to go back to listen. Although in that case the user cannot choose the leave the page group, she can still use the A**utoAnswerIntercom** option from the User Preferences menu to not to play any of the incoming pages

Note that when audio is received from the group, all your current calls will be interrupted in the standard way as if a new foreground call is added. When joining two groups, the audio from the two groups are mixed by the phone when received at the same time. The phone can mix up to two RTP streams in each page group.

Music On Hold (MOH)

The OBi1000 series have native support of MOH. You may configure a MOH server in the parameter ***Phone Settings – Calling Features***::**MOHServiceNumber**. The MOH Server number can be an external SIP or OBiTALK addressable device, such as pp(ob600559558) or sp3(moh-server). The expected behavior of the MOH Server device is such that when called, it automatically answers the call and starts streaming audio to the calling device.

The OBi1000 has a built-in MOH Server that can be invoked by specifying: `an({prompt})` (where an stands for the internal announcement server), and prompt is a specification of the source to play. The devices includes a Jazz track that can be used for music on hold. To enable the built-in track, specify the MOHServiceNumber as `an(jazz)`

Premium Call Features

Busy Lamp Field (BLF)

(SIP/SP only.) BLF is a common collaborative feature for a user to monitor the extensions of other users from their phone. BLF is a feature of the monitoring phone, The monitored extension usually does not do anything special to be monitored. The detection and notification of events occurring at the monitored extensions are carried out by the soft-switch.

When BLF is supported by a specific SP service on the OBi1000, the BLF function can be assigned to a feature key bound to that service to monitor an extension in the context of that SP service (That is another extension on the same hosted voice service or PABX system). We refer to a feature key that is assigned the BLF function as a BLF key. One BLF Key monitors one extension only. The administrator can assign as many BLF Keys as there are feature keys available, to monitor as many extensions. The following table lists the typical events that are monitored with a BLF Key. The table also shows the operation that can be invoked when the respective event occurs, and the LED pattern indicating that particular event.

BLF Event	Description	Available Operation	Icon	LED Pattern
Ringing	Monitored extension is ringing	Answer the call. This operation is known as *Directed Call Pickup*		Fast blinking red
Busy	Monitored extension is on a call or making a call	Barge In to either a) fully participate in the call or conference, or b) listen to the conversation without talking, or c) listen to the conversation but talk only to the monitored extension (a.k.a. coaching or whispering). There can be other variations along these lines.		Steady red
Holding	Monitored extension is holding one or more calls	Resume and take over the call		Slow blinking red
Call Parked	A call is parked *against* the monitored extension	Pickup the call from the parking slot		Periodic short blink in red
Idle	No calls on the monitored extension			Off
Offline	Status of the monitored station is unknown			Steady amber

Single Versus Multiple BLF Event Notification

There may be multiple events happening on the monitored extension for multiple concurrent calls (in various states). For example, the monitored extension may be on a connected call while an incoming call is ringing. Under these circumstances, some soft-switch implementations may only notify the monitoring phones of just one event that it

deems most appropriate, while some may notify all the concurrent call events. When an OBi1000 is notified with multiple call events on the same monitored extension, it selects the call event, called the *primary event*, to update the BLF key LED and icon with according to the following priority:

- Ringing (for one more incoming calls)
- Holding
- Call Parked
- Busy

To see all the call events, the user can invoke the corresponding Feature Key Item by pressing and holding down the key (for about 1s) until the item shows up on the screen. Once feature key item is selected, the screen shows a list of call events currently happening on the monitored extension. The user may then highlight the call to apply the desired operation on the call using the available soft keys.

Note that the level of details for each call event may not be the same for all soft switches. Some implementations may include the call peer's name and number information while others may only include the call states. The OBi1000 will show each call state as well as its call peer name and number if they are available.

BLF with Call Park Status

As described in the call park section, the OBi1000 supports two types of call park: a) Park a call against an extension (where each extension has a single parking slot can be identified by the same extension number), or b) Park a call in an available orbit out of many. In the first case, a soft-switch may include the call park status of the monitored extension in the BLF event notifications sent to the monitoring phones. At the time of writing, BroadSoft is the only soft-switch that is known to support this feature. As described above, the OBi1000 supports call-park status that is present in BLF notifications.

What Happens When BLF Key is Pressed

The OBI1000 reacts to a BLF key press based on the primary event. If it is ringing, the key press will trigger a Directed Call Pickup request sent to the soft-switch to answer the call on behalf of the monitored station, if the operation is available on the soft-switch and enabled in the phone configuration. Otherwise, the OBi1000 handles a BLF key press the same way as pressing a speed dial by calling the monitored extension. Other operations: Barge In, Call Pickup, and Resume, can only be invoked using soft keys from within the Feature Key Item screen of the BLF key.

BLF Operation: Speed Dial

When used as a speed dial, the OBi1000 determines the number to call the monitored extension according to the following logic, based on attributes specified in the Number parameter of the BLF key:

- If the optional `spd` attribute is specified, call that attribute, else
- If the optional `ext` attribute is specified, call that attribute, else
- Call the {*userid*} attribute, which MUST be specified for a BLF key

BLF Operation: Directed Call Pickup

Directed Call Pickup can be done in one of two ways: a) Feature Code or b) INVITE+Repleaces. The method to use is configured under the ITSP Profile of the SP Service that is bound to the BLF key.

For the Feature Code method, the Feature Code to use for Directed Call Pickup is also configured under the same ITSP Profile; the OBi1000 sends a normal INVITE to the number formed by concatenating the Feature Code with the number of the monitored extension (the `ext` attribute of the **Number** parameter, if it exists or else the {*userid*} attribute). (BroadSoft, FreePBX)

For the INVITE+Replaces method, the OBi1000 sends an INVITE with a Replaces header that identifies to the soft-switch the ringing call to pick up from the monitored extension. (MetaSwitch)

BLF Operation: Barge In

When the monitored extension is on a connected call, the monitoring phone may barge in to join the call, if the soft-switch supports the operation and if the feature is enabled in the phone configuration. This operation requires the specification of a barge-in feature code that can be configured under the ITSP Profile of the SP service that is bound to the BLF key. To barge in, the OBi1000 sends a normal INVITE to the number formed by concatenating the barge-in feature code with the number of the monitored extension (the `ext` attribute of the **Number** parameter, if exists or else the {*userid*} attribute). (BroadSoft, FreePBX)

BLF Operation: Call Pickup

When a call is parked against the monitored extension, the monitoring phone can pick up the parked call by sending a normal INVITE to the number formed by concatenating a call pickup feature code with the number of the monitored extension (the `ext` attribute of the **Number** parameter, if exists or else the {*userid*} attribute). The call pickup feature code can be configured under the ITSP Profile of the SP service that is bound to the BLF key. (BroadSoft)

BLF Operation: Resume

The Resume operation is intended to resume (and take over) a holding call on the monitored extension. At the time of writing, there is no known soft-switch that supports this operation.

BLF Configuration

Select a feature key **Key** N that may be a line key, side car key or programmable key to configure the following parameters:

Parameter Group	Parameter	Description
Key N ***N = 1,2,3…***	**Function**	Must be `Busy Lamp Field`
Key N ***N = 1,2,3…***	**Service**	Select the SP service that provides this function and signalling method must be SIP. For example `SP1`
Key N ***N = 1,2,3…***	**Number**	General Syntax: [{*group*}`/`][{*userid*}][`?`][`;ext=`{*extension*}][`;ptt`][`;spd=`{*speed-dial*}] where - {*userid*} is the userid of the monitored extension; it may be a generic user name, a DID number or an internal extension number assigned to that extension - {*extension*} is the (callable) extension number assigned to the monitored extension if it is different from the userid of that extension. For example, an extension may have {*userid*} = `ObihaiUser2` and {*extension*}=`1002`. If the `ext` field is specified, then its value is used for speed dial, directed call pickup, barge in, and call pickup. Otherwise {*userid*} is used. In the case of BroadSoft, it is required to use the {*extension*} instead of the {*userid*} for such operations in this context. - if the `ptt` flag is specified, OBi1000 handles the call as a push-to-talk call, when the key press is interpreted as a speed dial to the monitored extension - {*speed-dial*} can be used to specified an alternative number to call when pressing the key as a speed dial. This can be useful for adding some prefix digits to force certain behaivor of the call. For instance adding a prefix 8 to the extension tells the called party to auto-answer in FreeSwitch; this when combined with the ptt flag provides the familiar Intercom user experience
Key N ***N = 1,2,3…***	**Name**	A nickname or user-friendly name for the monitored extension. It can be used by the phone in some GUI elements

Key N *N = 1,2,3…*	**Group**	Not used
ITSP Profile X – SIP – Feature Configuration *X = A-F*	**X_BLFSubscribeExpires**	BLF implementation relies on the (SIP) subscription to the dialog event package of the monitored resource. This value specifies in seconds the expires value of such subscription. The phone will renew the subscription well before the nominal expiration time.
ITSP Profile X – SIP – Feature Configuration *X = A-F*	**X_DirectedCallPickupMethod**	Specifies the method to use for directed call pickup (i.e., answer the call that is ringing on the monitored extension). The choices are: - Feature Code: Phone dials the number that is formed by prefixing the number of the monitored extension with Directed Call Pickup Feature Code. This method is used by BroadSoft, FreePBX, and FreeSwitch - INVITE+Replaces: Phone sends an INVITE with a Replaces header that specifies to the server the call that it wishes to take over. This method is used by MetaSwitch
ITSP Profile X – SIP – Feature Codes *x = A-F*	**DirectedCallPickup**	Feature Code for directed call pickup.
ITSP Profile X – SIP – Feature Codes *x = A-F*	**CallPickup**	Feature Code for call pickup
ITSP Profile X – SIP – Feature Codes *X = A-F*	**BargeIn**	Feature Code for barge in
SPn Service – BLF Features *n = 1-6*	**X_BlfBargeIn**	Enable Barge In option for BLF keys associated with this SP*n* service
SPn Service – BLF Features *n = 1-6*	**X_BlfCallPickup**	Enable Call Pickup option for BLF keys associated with this SP*n* service
SPn Service – BLF Features *n=1-6*	**X_BlfDirectedCallPickup**	Enable Directed Call Pickup option for BLF keys associated with this SP*n* service

Floating BLF Key Assignment

It is possible to reserve a block of BLF keys to members of a group of extensions without hard-coding the extension for each reserved key. The idea is to SUBSCRIBE to a single resource that includes a list of extensions to monitor, instead of subscribing to each individual extension. Upon receiving a NOTIFY from the server with the list of extensions, the OBi assigns each extension to a BLF key reserved to the subscribed group using an internal algorithm. To reserve a BLF key to a member of the group, specify the Number parameter of a BLF key as: **Number** = {*group-name*}/

For example:

Number = `sales-team/`

The slash (/) at the end indicates the key is reserved to a member of the group named `sales-team`. You may also specify a preferred extension to use that key with the syntax: **Number** = {*group-name*}/{*preferred-extension*}?

For example:

Number = `sales-team/1003?`

The question-mark (?) at the end indicates that 1003 is merely a suggestion and will be used to monitor extension only if it is present in the list of extensions returned by the server in a NOTIFY message body. If 1003 is not in the extension list from the server, the OBi1000 is free to assign this reserved BLF to another key when it is running out of free keys to assign. So, if all the BLF keys are floating without specifying they are a suggestion, the order of extension assignment follows precisely the order of the extensions in the list received from the server.

Note that the block of BLF keys reserved for a group does not necessarily come from a continuous range of feature keys - the keys can be any combination of line keys, side car keys, and programmable keys.

SIP for BLF

For each BLF extension that does not belong to any group, the OBi1000 subscribes to the dialog event package for each extension in the context of the SP service specified in the **Service** parameter of the corresponding feature key. For extensions belonging to the same group, the OBi only maintains one subscription to the group-name and none for the individual extensions in the group. In the subscribe request, the OBi indicates (in an Accept header) support for the following content-types in NOTIFY message bodies to be returned by the server

- application/multipart/related
- application/rlmi+xml
- application/dialog-info+xml

For floating key assignment, the OBi expects the NOTIFY message body to include a resource list (Content-Type: application/rlmi+xml) that is compatible with RFC4662. Here is an example with two extensions specified in the resource list:

```
<?xml version="1.0" encoding="UTF-8"?>
<list xmlns="urn:ietf:params:xml:ns:rlmi" uri="sip:sales-team@as.broadworks.net"
  version="0" fullState="true">
  <resource uri="sip:ObihaiUser2@as.broadworks.net">
    <name>Obihai User2 </name>
    <instance id="DNAbROacM9" state="active" cid="7jTC13@broadworks"/>
  </resource>
  <resource uri="sip:ObihaiUser9053@as.broadworks.net">
    <name>Test User</name>
    <instance id="MT8FRckGPc" state="active" cid="cJ489p@broadworks"/>
  </resource>
</list>
```

To notify the call states of a monitored extension, the OBi expects the NOTIFY message body to include a dialog-info XML (Content-Type: dialog-info+xml) that is compatible with RFC4235. Here is an example:

```
<?xml version="1.0" encoding="UTF-8"?>
<dialog-info xmlns="urn:ietf:params:xml:ns:dialog-info" version="1"
  state="full" entity="sip:ObihaiUser2@as.iop1.broadworks.net">
  <dialog id="Y2FsbGhhbGYtNjI4MjM4NzU6MA==" direction="recipient">
  <state>proceeding</state>
  <local>
    <identity display="ObihaiUser2 ObihaiUser2">
      sip:ObihaiUser2@as.iop1.broadworks.net
    </identity>
    <identity display="ObihaiUser2 ObihaiUser2">
      tel:+12404982562;ext=2562
    </identity>
  </local>
  <remote>
    <identity display="ObihaiUser1 ObihaiUser1">
      sip:2561@as.iop1.broadworks.net;user=phone
    </identity>
  </remote>
</dialog></dialog-info>
```

When the soft-switch is capable of notifying call park status for the monitored extension, it is expected the status is reported inside a dialog-info XML as well. Here is an example:

```
<?xml version="1.0" encoding="UTF-8"?>
<dialog-info xmlns="urn:ietf:params:xml:ns:dialog-info"
  xmlns:sa="urn:ietf:params:xml:ns:sa-dialog-info"
  xmlns:bw="http://schema.broadsoft.com/callpark"
  version="3" state="partial"
  entity="sip:north00@txasdev87.net">
  <dialog id="Y2FsbGhhbGYtMzM6MA==">
    <state>confirmed</state>
    <bw:callpark>
      <bw:parked>
        <identity display="Alice south">
          sip:9726987601@as.bw.com;user=phone
```

```
        </identity>
      </bw:parked>
    </bw:callpark>
  </dialog>
</dialog-info>
```

BLF With OBiTALK Service

While most BLF implementation requires the soft switch to act as the state aggregator, OBiTALK service offers a peer-to-peer version of BLF that can be quite handy. Every OBi1000 Phone can notify its phone status to a configured OBi number that is configure in the ***OBiTALK Service – Calling Features***::**CallStatusAggregator** parameter. The notified phone monitors each notifying phone with a BLF function key that is configured with the Voice Service field set to OBiTALK and number set to the monitored 9-digit OBi number.

The following phone status can be monitored:

- Offline
- Idle (no calls)
- Ringing (at least 1 call ringing)
- Holding (at least 1 call holding)
- Connected (at 1 call connected)

On the monitoring phone, the following operations can be applied to the monitored OBi Number:

- Call (Call the monitored OBi Number)
- (Directed) Pick Up (Pick up the oldest ringing call on the monitored phone)
- Barge-In (Barge in a connected call on the monitored phone)
- Coach (Coach the monitored phone on a connected call)

Note: The call on the monitored phone to be picked up, barged in, or coached does not need to be on OBiTALK service. If the call is not on OBiTALK service, the monitored phone briges the call with the OBiTALK call leg with the monitoring phone that initiates the operation.

Call Park and Call Pickup

(SIP/SP only.) Call Park and Call Pickup (or Call Retrieval) are complementary operations not unlike call hold and call resume, except the pickup (vs. resume) operation can be carried out on extensions other than the one that parks (vs. holds) the call. Generally speaking, a “parking lot” for calls with many slots can be setup at the Soft Switch such that a call may be parked at an unoccupied slot from one extension and picked up by the same or another extension from the same slot. In some implementations the parking slot is referred to as a *park orbit*.

Each parking slot can hold one call and is uniquely identified with a numeric ID (the slot-id or orbit), such as 001, 1234, 88912, etc. While a user is talking to someone on the phone, they can choose to park the call at an unoccupied slot, and later they (or someone else) can pick up the call from the same slot. From a usability standpoint, it may be inconvenient for the user to first find out which parking slots are available before parking a call. A simple solution is to let the system pick an available slot to park the call, in a certain pre-defined range when requested by a certain user. The latter case can be refereed to as “valet parking”, while the former “self parking”. The problem with valet parking is that the system still needs to communicate back to the user the parking slot ID that has been chosen, so that the call may be picked up later from that slot. In some systems, this is done by showing the parking slot ID on the phone screen, which obviously only works for phones with a display. Another strategy of choosing a parking slot is to use the parking phone's extension number as the parking slot identifier, for easy memorization. You can think of each extension as having an associated parking slot, such that one call can be *parked against one extension* that has this resource enabled. When the user attempts to park a call, you can park against the current extension the call is on by default, or against another extension by explicitly entering the target extension. Similarly you can pick up a call that is parked against your current extension, or against another extension by explicitly entering the target extension. In applications where most users normally park no more than one call at a time, a user can just park against his own extension; thus avoiding parking slot collisions. With this approach, some BLF implementations also include parking slot monitoring when monitoring an extension.

Call Park Methods

The OBi1000 supports both the types of call park described above. The method to use can be set independently for each SP service, under the ITSP Profile bound to that SP service, with the parameter ***ITSP Profile–SIP***::**X_CallParkMethod**. The `Feature Code` method must be used for the *Park-Against-An-Extension* method, while the `REFER` must be used for the *Park-In-An-Orbit* method.

With the Feature Code method, the call park feature code must be specified under the same ITSP Profile using the parameter ***ITSP Profile–SIP-Feature Codes***::**Park**. When user presses the "Park" soft key for a highlighted call on screen, OBi1000 parks the call (in Holding or Connected state) against the local extension of the underlying SP service by sending a normal INVITE to the number formed by concatenating the call park feature code with the extension number of the local extension. This INVITE (to park) is sent on the same SP service that the call to be parked is on. Note that in the context of parking a call, the local extension of the SP service is taken from the parameter ***SPn Service***::**X_MyExtension**, if it is specified. Otherwise, the normal userid of the SP service is used. The phone GUI does not have a soft key to park a call against an arbitrary extension - to do that, the user must make another call by dialing the call park feature code followed by the extension to park against. The burden on the user can be alleviated by defining a number of speed dials with the feature code+extension pre-configured so the user can simply press the speed dial key to park the call in one of the pre-defined extensions.

With the REFER method, the OBi1000 parks the call in an orbit by SIP-referring the peer of the call to be parked to the number that is the orbit to be parked at. This operation is equivalent to that of a blind transfer to a special extension. When user presses the "Park" soft key of a highlighted call on screen, the OBi1000 pops up a dialog box to ask the user to enter the orbit number to park the call at. Alternatively, user can simply press the Call Park Monitor feature key of an unoccupied park orbit to park the call with a single key press. The *Call Park Monitor* feature key is described in the next section.

Note that the "Park" soft key will not be seen unless the parameter ***SPn Service–Network Provided Services***::**CallPark** is enabled.

Call Park Monitor and Call Pickup Methods

Generally speaking, when a call is parked-against-an-extension with a feature code, the user can pick it up by dialing the call pickup feature code followed by the extension number the call is parked against. To the phone this is just an ordinary call and it does not need to know, or interpret, the call pickup feature code; the user has the full responsibility of dialing the feature code and the parked-against extension explicitly and correctly. Similarly when a call is parked-in-an-orbit and if the soft-switch has a way to allow calls to be picked up from an orbit that is equivalent to making a call to a special number, users can make those calls without any special support from the phone. There are scenarios when the OBi1000 can monitor if certain extensions have calls parked against them, or if certain park orbits are holding calls.

For calls parked-against-an-extension, OBi1000 may monitor call park status by directly (SIP) subscribe to an extension's call park status (by enabling the option ***SPn Service–Network Provided Services***::**CallParkStatus)**, or indirectly through BLF monitoring of that extension (see the section *Busy Lamp Field*). Note that OBi1000 supports direct subscription to call park status (in the context an SP service) of the underlying SP's own extension only, but not an arbitrary extension. Note also that if the underlying SP is a shared line, then the direct subscription to call park status of that extension is not necessary if the soft-switch already includes call park status in the notifications of the shared line status to the phone. The OBi1000 supports call park status subscription by subscribing to the *x-broadworks-callpark* event package, and expects notification that includes an x-broadworks-callpark-info XML document in the message boxy (Content-Type:application/x-broadworks-callpark-info+xml). Here is an example:

```
<?xml version="1.0" encoding="UTF-8"?>
<x-broadworks-callpark-info xmlns="http://schema.broadsoft.com/callpark">
  <callpark>
    <parked>
      <identity display="Alice south">
        sip:876601@as.bw.com;user=phone
      </identity>
    </parked>
  </callpark>
</x-broadworks-callpark-info>
```

In theory, with the park-against-an-extension method, one should be able to define a group of extensions to serve as a more generic parking lot and use BLF keys to monitor the call park status of each key. With the `spd` attribute of the **Number** parameter these BLF keys will include the call park feature code as a prefix to act as a short-cut to park a call in an empty slot.

For calls parked-in-an-orbit, the administrator can define a feature key with the function *Call Park Monitor* to monitor each orbit (one feature key per orbit; each feature monitors a different orbit number). The OBi1000 treats the Call Park Monitor function always exactly like the BLF function; it subscribes to the dialog event package for each key (or for a group of keys using the same group syntax in the **Number** parameter of the keys). It is however expected that the soft-switch either notifies no calls or one call in Ringing state per orbit, as summarized in the following table.

Call Park Monitor Status	Description	Available Operation	Icon	LED Pattern
One Call Ringing	A call is parked in this orbit	Pickup the call. This operation is done via *Directed Call Pickup*		Fast blinking red
No Calls	No call parked in this orbit	Park the call on this orbit		Off
Offline	Status of the park orbit is unknown			Steady amber

The configuration of Call Park Monitor feature key is similar to that of a BLF feature key, except that the {*userid*}, {*extension*} , and {*speed-dial*} attributes pertain to a park orbit instead of a real extension and the `ptt` flag is not applicable.

When the user presses the Call Park Monitor key, one of the following applies:

- If there is no call parked, the highlighted call on the screen is parked
- If there is a call parked, pickup the call

Call Park Ring

When there is one or more Call Park Monitor keys defined, the Call Park Ring can be enabled to play a short ring at regular interval as an indication that at least one of the monitored call park orbits has a call parked. This feature can be enabled using the boolean parameter ***Phone Settings – Calling Features***::**CallParkRingEnable**, which is `false` by default. There are two internal short ringtones that can be chosen for this ring: `ding` or `blop`. Which short ring to use, as well as the ring interval and total ring duration can be set in the parameter ***User Preferences***::CallParkRingtone which takes a string value of the following general format:

{*ring-tone-url*},duration={*d-value*},interval={*i-value*}

where

- {*ring-tone-url*} can be `ding` or `blop` or an external URL that starts with `http://` or `https://`. For best results, the ring tone for this should be about 1-2s long.
- {*d-value*} is the total duration in seconds to play the ring for, as long as there is a parked call. It must be greater than 0 or equal to 0 to mean play forever. The duration parameter is optional; default value is 60.
- {*i-value*} is the interval between the ring bursts in seconds. It must be greater than 0. The interval parameter is optional; default value is 10

To let phone user change the **CallParkRingEnable** option from the User Preferences GUI on the phone, add the `cpr` option in any one of the ***GUI Menus***::**PerferencesMenu***n** (*n* = 1 – 6). The default on-screen label for this option is Call Park Ring and the use can set the value to either Enable or Disable.

Notes:

- The call park ring will not start in any of the following situations:
  - Phone is ringing
  - Phone is off-hook, speakerphone is on, or headset is on
- Regardless of the duration setting, the call park ring will stop when any of the following happens:
  - No more calls parked in any of the monitored parking orbits
  - Phone starts ringing
  - Phone goes off-hook, or the speakerphone or headset is turned on
- If a new call is parked while the call park ring is playing, the duration timer restarts so it can potentially ring for a full duration for the newly added call

Tips: When call park ring is playing, an easy way to stop it (until a new call is parked) is by taking the phone off-hook or turning on the speakerphone or headset momentarily.

Shared Line and Shared Call Appearances (SCA)

(SIP/SP only.) A Shared Line is a service account or extension that is installed on a group of phones, such that if a sharing phone is using that extension, other sharing phones will be notified. There can be multiple simultaneous calls on a shared line. The maximum allowed simultaneous calls on a shared line should be a fix number. Each call on a shared line is called a Shared Call Appearance (SCA). If your phone has a shared line configured as one of the services, it should have as many Call Keys defined on the phones that are bound to that service. SCAs of a shared line are ordered with an index 1–*n* as SCA1, SCA2, SCA3, ... SCA*n*, where *n* is the maximum number of calls permitted for that line. On the OBi1000, the SCAs are assigned sequentially in ascending order according to the VLK index of the corresponding Call Key bound to the shared line. For example, suppose a shared line with 4 SCAs is configured on a phone with VLK1, VLK3, VLK7, and VLK8 assigned as the 4 Call Keys bound to that line, then the calls hosted on VLK1, VLK3, VLK7, and VLK8 are SCA1, SCA2, SCA3, and SCA4, respectively.

There are two common implementations of SCA, namely, the Call-Info Method and the SLA/BLA (dialog) method. Use the parameter ***ITSP Profile–SIP–Feature Configuration***::**X_ShareLineMethod** to select which method to use. It can be one of the following choices:

- `call-info`: This is the method used by BroadSoft. The phone subscribes to the *call-info* event package with the proxy server to receive notification of share call appearance state updates. This method also uses the line-seize event package for seizing a SCA before making a call

- `dialog;sla`: This is based on the Bridged Line Appearance draft (draft-anil-sipping-bla-02). The phone accepts subscription to the *dialog;sla* event package from the state-agent, and also subscribes back the same with the state-agent. The phone and state agent then exchange notifications of share call appearance state updates.

- `dialog;ma`: This is similar to the `dialog;sla` method, and is based on a more current version of the same draft (draft-anil-sipping-bla-04). The name of the event package name is changed to *dialog;ma*.

To designate an SP service as a shared line, enable the option ***SPn Service–Share line Features***::**X_ShareLine**. The ***SPn Service–Calling Features***::**MaxSessions** parameter should be set the value of the maximum number of call appearances allowed on the shared line. You must also have the same number Call Appearance feature keys defined to bind to this SP service. A Call Appearance key bound to a shared line behaves similarly to one that is bound to a private

line, except that that when the (private) call state is idle (i.e. the call appearance is not being used by this phone), the SCA state is shown. The SCA state (or shared call state) indicates the call state on the sharing phone that is using that call appearance, and is communicated to this phone by soft-switch via (SIP) notification. The following SCA states are supported by the OBi1000:

SCA State	Description	Available Operation	Icon	LED Pattern
Seized	A sharing phone has seized the SCA to make an outgoing call			Blinking in red (30 ms on/30 ms off)
Trying	A sharing phone is trying an outgoing call			Steady red
Proceeding	A sharing phone is making an outgoing call and the called party is ringing			Steady red
Connected	A sharing phone is on a connected call with that SCA	Barge In to monitor the conversation only or to fully participate in the conversation		Steady red
Held	A sharing phone is on a call with that SCA and has placed the call on hold	Resume and take over the call		Slow blinking in red
Private Held	A sharing phone is on a call with that SCA and has placed the call on private hold			Slow blinking in red (100 ms on/100 ms off)
Call Parked	A call is parked against the extension	Pickup the parked call		Blinking in red (50 ms on/50 ms off)
Idle	None of the sharing phones is using the SCA	Seize the SCA to make an outgoing call		Off
Error	Protocol or network error			Steady amber

SCA implementations are based on subscribe/notify framework. With the call-info method, the subscription expires value to the call-info and line-seize events are set respectively in **X_CallInfoSubscribesExpires** and **X_LineSeizeSubscribeExpires** (under ***ITSP Profile X – SIP***). With the dialog;ma and dialog;sla methods, the subscription expires value to the respective event is controlled by the **X_DialogSubscribesExpires** parameter.

Line Seize

With any Shared line design, each sharing phone should perform a proper line-seize before attempting to make a call on a SCA. This is make sure the system will work correctly when multiple phones are trying to make calls at the same time.

With the call-info method, a phone seizes a SCA by subscribing to the line-seize event package for that SCA, with a short expires value. If successful, the phone will receive a 200 class response for the subscribe method, and also a NOTIFY to indicate that the subscription is active and what the actual expires value of the subscription allowed by the server.

With the *dialog;sla* or the *dialog;ma* method, a phone seizes a SCA by sending a NOTIFY to the state-agent with the state of the requested SCA set to “trying”. If successful, the phone will receive a 200 class response for the NOTIFY request. Typically the state-agent will shorten the subscription interval with the phone who is owning the SCA in order to detect quickly if the phone has crashed or encountered network issues, etc.

What Happens When a Call Appearance Key is Pressed

- If the SCA is Idle, the phone will try to seize the line with a new line seize operation. User will hear Dialtone only if the subscription is successful
- If the SCA is Holding, the phone will try to resume the call
- If the SCA is Connected, the phone will try to barge-in
- If the SCA has a call parked (against its extension), the phone will try to pick-up
- For other SCA states, the phone will do nothing

Buddy List

A Buddy List is a contact list with presence information incorporated. The OBi1000 supports Buddy Lists based on the XMPP standard. This feature is enabled with the option ***SPn Service – Network Provided Services***::**BuddyList**. Note that the service can be enabled for each SP service independently, using SIP or Google Voice. If the SP service is a Google Voice service, then the buddy list is based on the same Gmail account and no further configuration is required. If the SP service is a SIP service, the following additional parameters are required:

Parameter Group	Parameter	Description
SPn Service – SIP Credentials	**X_XmppDomain**	The XMPP server domain name. The phone will resolve this name as DNS A Record only. For example: `imp1iop1.broadsoft.com`
SPn Service – SIP Credentials	**X_XmppUserName**	The username for authentication to the XMPP server. For example: `ObihaiTester@imp1iop1.broadsoft.com`
SPn Service – SIP Credentials	**X_XmppPassword**	The password for authentication to the XMPP server

You and your buddies must be using the same XMPP service (such as Gmail) in order to see the presence and status of each other (there is no XMPP federation). Each entry in the buddy list consists of the following information:

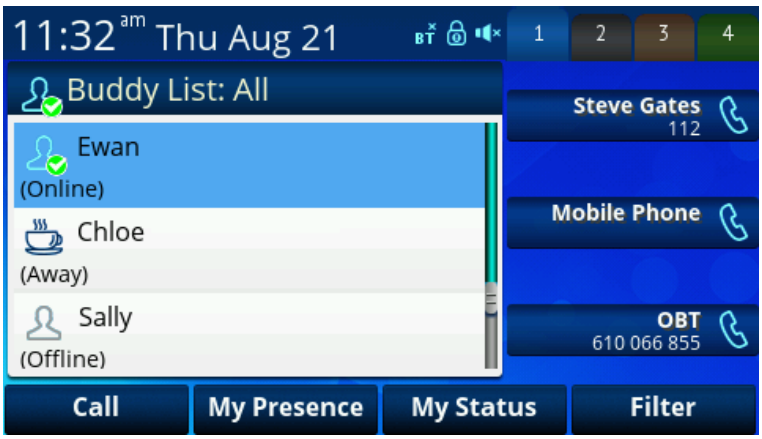
- JID (Jabber ID): Every user of an XMPP service is identified with a unique User ID called Jabber ID for historical reason. It is in the form of an email address such as: `bbking@gmail.com`
- Display Name (optional): The display name of the user, such as: `Benjamin B. King`
- Phone Number (optional): This may be a public number or an internal extension if the XMPP service is an internal service deployed within an enterprise for example
- Presence, which may take one of the following values:

Presence	Description	Icon
`Offline`	The user is offline. This is also known as “invisible” in some XMPP clients	
`Online`	User is online. This is also known as “available” in some XMPP clients	
`DND`	The user is busy and does want to be disturbed. This is also known as “busy” in some XMPP clients	
`Away`	The user has stepped away from the computer momentarily	
`Extended Away`	This user has stepped away from the computer and may take a while to return	

- Status (optional): An arbitrary text string usually entered by the user to provide further details of his current state. For example: `Having my lunch break`, `Back at 1 pm`, etc.

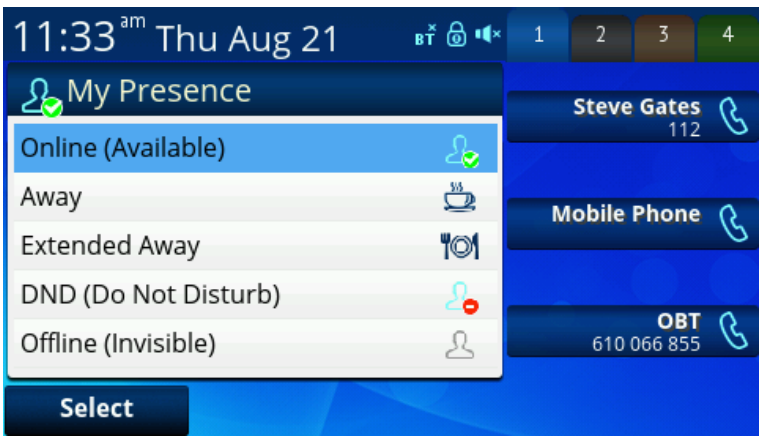
The Buddy List App is launched from the Main menu on the Home Screen. For the Buddy List App to show, it must be enabled for a service under ***SPn Service – Network Provided Services***::**BuddyList** to display contact images, ensure you also enable the QueryVcard option further down the page. The picture below shows an example of a Buddy List.

A Buddy List filtered by the "All" Group Filter



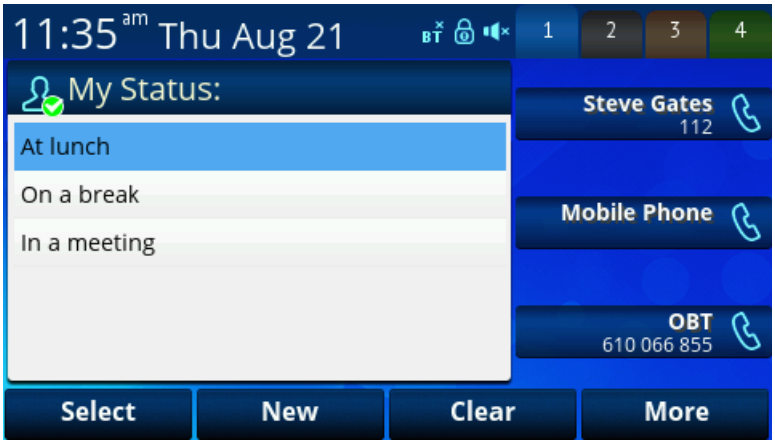
In addition to showing the buddy list as described above, the user has the option to set their own presence and status as seen by other users of the same XMPP service. The user sets their own presence by pressing the **MyPresence** soft key; the default value is `online`. An example screen after pressing **MyPresence** is shown below. The current Presence setting is shown as an icon displayed in the title area of the screen. The value of the user selectable presence is stored in the parameter ***SPn Service***::**MyPresence**.

Setting my presence from the GUI



The user sets their own status by pressing the **MyStatus** soft key; the default status is {blank}. Below is an example of the screen shown after pressing **MyStatus**. It shows a history of status messages that the user has entered before, with the current user status message shown in the title area of the screen. The user can edit or remove an old status message, or add new ones. The user can also select a different message from the list or clear the current status by pressing the **Clear** soft key. The user selected status value is stored in the parameter ***SPn Service***::**MyStatus**. The user may add new status values or remove old ones (using the **New** and **Remove** soft key respectively); all the status values are available to all SP services and are stored as a comma separated list of phrases in the parameter ***PhoneSettings***::**MyStatusHistory**.

Setting my status from the GUI



Note that whenever the phone is in use (i.e., off-hook), the phone automatically sets the user status to "`On the phone`", with presence = `DND`. It restores to the last user set presence and status when the phone returns to idle.

Expanded Buddy List and Groups

If the XMPP server is from BroadSoft, the OBi1000 supports expanded buddy list and groups.

An expanded Buddy List may include basic contacts that do not have presence information. For those contacts, you will not see a presence icon in the respective entry in the buddy list display. This is just a matter of convenience so that you can have all your contacts consolidated in one list, with or without presence information.

A Buddy List may be divided into groups such as "Friends", "Co-workers", etc. A contact may belong to more than one group or no group at all. By default all contacts in a Buddy List are in the "All" group. Most Buddy List implementation also allows the user to tag a contact as "Favorite" (sometimes called "Starred"). The phone treated this as the "Favorite" Group. Additional groups may be defined by the user.

Buddy List Management

Buddy list management refers to operations like

- adding a contact
- removing a contact
- adding a group
- removing a group
- adding contacts to a group
- removing contacts from a group
- tagging or un-tagging a contact as a Favorite
- accepting presence subscriptions (i.e. invites) from other users

The OBi1000 does not support any of these management operations on the buddy list from the phone GUI. Users are expected to perform such operations using compatible client software that runs on a PC, tablet, smart-phone, etc. For example Google Voice users can use a Gmail client to change the buddy list. BroadSoft users can use the *BroadTouch Business Communicator* client to do the same. The phone will pick up the changes to the buddy list from the XMPP account automatically.

Presence Monitor

You can configure a feature key with the function `Presence Monitor` to monitor the presence of a buddy in a buddy list. Each presence monitor key monitors exactly one buddy. You can configure as many Presence Monitor keys

as you need, but do not allocate the same buddy to more than one key. The following parameters are required for configuring a presence monitor key:

Function = `Presence Monitor`

Service = The SP Service where the XMPP service is offered, such as `SP3`

Number = The JID of the buddy to be monitored; this typically looks like an email address, such as: `kkytte@gmail.com`. Only the userid portion of the JID is needed. You may specify `..` at the end such that a partial match of the JID is sufficient to find the buddy in the buddy list. For example, if the full JID of the buddy is: *abcd-12345@gmail.com*, you may specify `abcd..` for the Number field to monitor this buddy, provided there is no other JID in the buddy list that starts with abcd.

Call Recording Controls

(SIP/BroadSoft.) The OBi1000 supports the call recording functions available with a BroadSoft application server, by providing the controls for call recording during a call. This feature can be enabled on a per SP service basis by enabling the option ***SPn Service – Network Provided Services***::**CallRecording**. When the feature is enabled, the phone shows one of the following call recording states in the call items of the Calls App:

Call Recording State	Description	Available Soft Key Options	Icon
off	Recording has been turned off	Rec.Start	■
on	Recording has been turned on	Rec.Stop, Rec.Pause	●
paused	Recording has been turned on but paused at the moment	Rec.Resume	▮▮
na	Feature is not enabled		

The soft key options for recording controls options are available only when the call is in the Connected or the Holding state.

Hold and Talk Event Package

(SIP Only.) The OBi1000 supports unsolicited notification of the *hold* and *talk* event package in the context of an SP service. Note that these notifications are sent outside of any dialogs and are not dialog specific. When the phone receives a hold event notification, it holds all the connected calls on the underlying service. When the phone receives a talk event notification, it answers all the incoming calls and resumes all the holding calls on the underlying service.

There is no configuration for these features.

Advice of Charges (AOC)

(SP/SIP Only.) The OBi1000 accepts contents of the *Content-Type: application/vnd.etsi.aoc+xml* in the message body of an INFO or BYE request, or 2xx response to a BYE request. It parses the charges information and displays them on the screen in the corresponding call item of the Calls App. The final charges information received with a BYE request or 2xx response to a BYE request is displayed in a separate pop up window at the end of the call.

There is no configuration for this feature.

BroadSoft Call Center Features

(SIP only.) This is a suite of features to support call center applications with a BroadSoft soft-switch.

Disposition Code

A disposition code can be entered by an agent for the current call that is still ongoing or for the last call that has just ended. For the first case, the agent selects the **Dispose Code** soft key that is available when the call is in the connected state. The agent then enters the code and submits it while talking to the caller. For the latter case, the agent can press the feature key that has been assigned the `Disposition Code` function right after the call, then enter and submit the code.

To use this feature on the phone, you must enable the option ***SPn Service – Network Provided Services***::**DispositionCode**. The option only applies to calls on the same SP service.

Customer Originated Call Trace

The agent can start a call trace during the call by pressing the **Trace** soft key when the call is in the Connected state; the actual call trace function is executed entirely on the soft-switch after invocation from the phone. To use this function on the phone, you must enable the option ***SPn Service – Network Provided Services***::**CallTrace**. The option only applies to calls on the same SP service.

To start a call trace outside the context of a call, the administrator can define a speed dial feature key with the following parameters:

Function = `Speed Dial`

Number = `customer-originated-trace`

Server = The SP Service to invoke the Call Trace operation with, such as `SP3`

Escalation

The agent can escalate the current call to a specific supervisor or the predefined default supervisor by selecting the **Escalate** soft key that is available when the call is in the Connected State. After pressing the key, the agent has a chance to enter a specific supervisor's extension, or skip that to use the default supervisor's extension. To use this feature on the phone, you must enable the option ***SPn Service – Network Provided Services***::**Escalation**. The option only applies to calls on the same SP service.

Call Center Information

When the soft-switch sends an incoming call to the phone, it may include some basic information about the call center where the call is coming from. Such information, if available, is displayed by the phone with the Ring Alert message for the incoming call. The following information can be displayed:

- Call Center name
- Call Center user ID
- Average waiting time
- Number of calls in the queue

This behavior as described above does not have any configuration.

In addition, the phone can proactively subscribe to the status of the call center that a SP service belongs to. This feature is enabled with the option ***SPn Service – Network Provided Services***::**CallCenter**. The call center status information can be viewed at any time on the phone by going through the Net Services App (from the phone's main menu).

BroadSoft Guest Login/Logout (Hoteling)

(SIP only.) This feature is also known as Hoteling (consult BroadSoft documentation on how to administer this feature on the server). The phone may be set up to be used temporarily by a guest, such as a visiting employee or temp worker in a hot desk environment. To use this feature on the phone, the option ***SPn Service – Network Provided Services***::**Hoteling** must be enabled. If enabled, the phone starts a subscription to the x-broadworks-hoteling event package in the context of a SP service, right after the first successful registration with the SIP Proxy Server. The expires value of this subscription can be set with the parameter ***ITSP Profile – SIP – Feature Configuration***::**X_BWHotelingSubscribeExpires**.

There are two ways to access the guest login/logout function:

- Within the Net Services App, select the SP service, and then select the Hoteling option
- Define a feature key with the function `Hoteling` (but no more than one hoteling feature key per SP service).

The LED and the icon of the key reflect the current guest login state, as shown in the following table:

Guest Login State	Description	Icon	LED
Guest Logged In	A guest has successfully login. The login extension is shown on the screen		Solid green
Guest Logged Out	No guest login		Off
Protocol error	Problem reaching the server or subscribing to the service		Solid orange

When trying to login, the guest will be first prompted to enter the guest extension. After submitting a valid guest extension, the guest will be further prompted to enter the corresponding password. Once the password is accepted by the server, the phone screen will show the guest extension in the Call Appearance and Line Monitor keys that are bound to the SP service. The guest may logout by pressing the hoteling feature key again once, or the server may logout the guest remotely; the screen will be updated automatically according to the updated guest login state.

Emergency Calls

The administrator can define one or more numbers as emergency numbers by adding the prefix EM# to those numbers using the ***Phone Settings***::**DigitMap** and a corresponding rule in ***Phone Settings***::**OutboundCallRoute** to route those calls to a specific voice service to handle the call. The following example defines an emergency number 911 and routes the call to go out from SP1 when the number is dialed:

DigitMap = `(<EM#>911|`*other rules …*`)`

OutboundCallRoute = `{('EM'#xx.):sp1},` *other rules …*

The phone detects that you are calling an emergency number if the number to call has the prefix EM# after applying the phone digit map on the dialed number. It then applies the emergency call treatment to that call for the duration of the call:

- You cannot hold or end the call; only the remote party can end it
- You can start the call with the headset; but you cannot switch to use a headset subsequently after the call is started. You can only switch between the handset and speakerphone
- You cannot start or resume any other calls
- You cannot press the Home or Cancel key to get to the Home screen to start another App
- Call waiting disabled
- All the feature keys are disabled

Call Diversion History

(SIP Only). The soft-switch may keep a history of Diversion headers for each call forward transaction as it tries to ring an extension. A Diversion header is added each time the called extension redirects (or diverts) the call. When an INVITE message arrives at the OBi1000 after a series of redirections, it may include a history of all the diversion headers (with the latest one appearing first). The OBi1000 will show the call diversion history in the call item of the Calls App, if available.

There is no configuration for this feature.

BroadSoft AS-Feature-Event Features

(SIP Only.) This is a collection of network-provided (i.e. soft-switch-provided) features available on a BroadSoft Application Server, the settings of which the user can view and change from the phone GUI. Note that these network-provided features are configured and executed in the context of a single SP service. To allow any of the network-provided features listed below to be viewable and changeable from the phone, you must enable the option ***SPn Service – Calling Fetures***::**X_ASFeatureEventSubscribe** and must also enable the individual network-provided feature whichever you want to allow users to access from the phone. Note that the features themselves are executed entirely on the server and the settings of the features are stored on the server. The phone displays the values of the settings as stored on the server (not the ones entered and submitted by user, which may or may not be acceptable by the server).

The as-feature is based on the SIP subscribe/notify framework; the expires value of the subscription dialog (initiated by the phone per SP service with the feature enabled) can be set using the parameter ***ITSP Profile X–Feature Configuration***::**X_ASFeatureEventSubscribeExpires**. On each subscribe request sent by the phone, the server returns a NOTIFY that refreshes the current settings of all the soft-switch-provided features listed below (whichever is enabled or made available on the server). When a setting is changed, the server also updates the phone with a NOTIFY that specifies the latest settings of just the affected features.

Some of the network-provided features can be accessed from the phone via special feature key functions (such as Do Not Disturb) or special soft keys (such as Call Forward All), while all enabled network-provided features can be accessed via the Net Services App of the Main Menu.

Call Forward All

CallForwardAll is synonymous to CallForwardUnconditional in this document. This feature, if enabled, will let the soft-switch forward all calls to the configured forwarding number unconditionally. The functionality provided by this feature is similar to that of the CallForwardUnconditional feature provided natively by the phone (per line); the administrator is advised to disable the native version when using the network-provided version to avoid ambiguity. To make the setting of this network-provided service viewable and changeable from the phone GUI, you must enable the option ***SPn Service – Network Provided Services***::**CallForwardAlways**. In addition to going through the Net Services app, the following methods of access are available:

- Feature Key: Define a feature key with the function Call Forward. Set the Service parameter of the key to the SP service that provides this feature
- Soft Key: The **Call Forward** soft key on the home screen of the phone can be mapped to the network provided CallForwardAll service on a specific SP service, by setting the parameter ***Phone Settings – User Preferences Settings***::**CallForwardUnconditionalFeatureProvider** to equal to that SP service
- User Preferences – Call Forward Setting: As a side effect of enabling the Soft Key option in the last item, the Call Forward Setting under User Preferences app on the GUI is also pointing to the same network provided CallForwardAlways feature

The user can enable/disable CallForwardAlways as well as setting a forwarding number; the settings are submitted and stored on the server.

Call Forward Busy

To make the setting of this network-provided service viewable and changeable from the phone, you must enable the option ***SPn Service – Network Provided Services***::**CallForwardBusy**. The functionality provided by this feature is similar to that of the CallForwardOnBusy feature that is available natively on the phone (per line). To use the version provided by the soft-switch, the administrator is advised to disable the native version to avoid ambiguity.

The only way of access from the phone is by going through the Net Services app. The phone allows the user to get and change the setting of this feature on the server; the setting is submitted to the server with a subscribe request and stored on the server.

Call Forward No Answer

To make the setting of this network-provided service viewable and changeable from the phone, you must enable the option ***SPn Service – Network Provided Services***::**CallForwardNoAnswer**. This feature is similar to the CallForwardOnNoAnswer feature that is available natively on the phone (per line). To use the version that is provided by the soft-switch, the administrator is advised to disable the native version to avoid ambiguity.

The only way of access from the phone is by going through the Net Services app. The user can enable/disable the feature, set the forwarding number, and also the number of rings before forwarding the call; the setting is submitted to the server with a subscribe request and stored on the server.

Do Not Disturb

To make the setting of this network provided service viewable and changeable from the phone, you must enable the option ***SPn Service – Network Provided Services***::**DoNotDistrub**. The functionality provided by this feature is similar to that of the DoNotDisturb feature that is available natively on the phone (per line). To use the version that is provided by the soft-switch, the administrator is advised to disable the native version to avoid ambiguity.

In addition to going through the Net Services app, the setting can also be accessed via:

- Feature Key: Define a feature key with the function `Do Not Disturb`. Set the Service parameter of the key to the SP service that provides this feature

User can enable/disable this feature from the GUI; the setting is submitted to the server with a subscribe request and stored on the server.

ACD Agent State

ACD stands for Automated Call Distribution and is the primary way a call-center distributes calls among a number of agents. Normally the ACD controller only sends a new call to an agent who is in the "Available" state. An agent typically "Signs On" when they arrive at work and then "Signs Off" when done for the day (or taking a very long break). The agent may at any time tag themself as "Unavailable" when taking a break, or "Wrapping Up" when filing paperwork for the last call before taking another call. Through as-event subscription, the OBi1000 allows an agent to sign on, sign off, or change their availability states from the phone GUI. To make this feature available on the phone, you must enable the option ***SPn Service – Network Provided Services***::**ACDAgent**.

User can invoke this feature in two ways:

- Press the feature key with the function ACD Agent
- Within the Net Services App, select the SP service, and then select the ACD Sign On/Off item

With the ACD feature key, the agent can sign on by pressing the key once.

With the OBi1000, an agent can sign on/off and change its state from the phone GUI. The agent can set its state to to one of the following values:

- Available (to take new calls)
- Unavailable (to take new calls)
- Signed Off
- Wrapping Up (the last call)

While "Signed Off", the agent presses the key once to sign on and becomes "Available". While "Available", the agent presses the key once to become "Unavailable"; the agent must also enter one of the valid unavailable-reason codes (such as 11) that are defined by the Call Center admin. While "Unavailable" or "Wrapping Up", the agent presses the key once to become "Available".

Note that the agent cannot change their state to "Signed Off" or "Wrapping Up" directly by pressing the feature key. To change to these states, the agent must use the corresponding feature key menu item from the GUI (invoked by pressing and holding down the feature key), or some other means provided by the Soft Switch, such as a web portal for agents.

Security Classification

To make the Security Classification setting viewable and changeable from the phone, you must enable the option ***SPn Service – Network Provided Services***::**SecurityClass**. With that the phone lets the user to view or set the security levels for his extension. The currently available security levels that the user is allowed to choose are presented to the user on the screen. In addition to invoking this function by going through the Net Services App, the administrator can also define a feature key with the function `Security Class` as a shortcut to launch this function.

Executive Call Filter

To make the setting of the Executive Call Filter option viewable and changeable from the phone, you must enable the option ***SPn Service – Network Provided Services***::**Executive**. In addition to invoking the function through the Net Services App, the administrator can define a feature key with the function `Exec Filter On/Off` as a shortcut to turn on/off this setting; the LED color also reflects the current on/off status.

Executive Assistant

By enabling the option ***SPn Service – Network Provided Services***::**ExecutiveAssistant** = true, the phone makes the following settings of this feature available to user:

- From a list of Executives currently associated with the assistant, turn on/off the call filtering feature for any of the executives
- Enable/Disable the "Divert" option and set/modify the Phone Number to divert to

In addition to invoking the function through the Net Services App, the administrator can define a feature key with the function `Exec Assistant` as a shortcut to launch this option; the LED color also reflects the current Divert on/off status.

Call Recording Settings

The phone can extract the call recording settings from the as-event notifications to help determine if and which call recording controls to present to the user during a call, provided the ***SPn Service – Network Provided Services***::**CallRecording** is also enabled. Note that these call recording settings are not changeable via XSI.

BroadSoft XSI Features

This is a collection of features that is provided with a BroadSoft XSI Application Server. The OBi1000 makes XSI Features available per SP/SIP service. This allows the configuration of up to 6 independent sets of XSI services per phone - one per SP service. To execute any of the features listed in this section, the following parameters are required:

Parameter Group	Parameter	Description
ITSP Profile X – SIP	**X_XsiServer**	The XSI server hostname or IP address. Phone will attempt to resolve the hostname as DNS A Record only (i.e. DNS SRV lookup not supported here)
ITSP Profile X – SIP	**X_XsiServerPort**	The server port. If not specified (or 0), the default port is used (80 for HTTP and 443 for HTTPS)
ITSP Profile X – SIP	**X_XsiServerScheme**	Must be HTTP or HTTPS
SPn Service – SIP Credentials	**X_XsiUserName**	The username to authenticate to the XSI server with. If not specified (blank), the phone forms the user name as: {*sip-userid*}@{*sip-domain*} where {*sip-userid*} is the SIP Account User ID that is used for SIP Registration on the same SP service, and {*sip-domain*} is the domain name that is used for SIP Registration on the same SP service.

SPn Service – SIP Credentials	**X_XsiPassword**	The password to authenticate to the XSI server with. If not specified (blank), the same password for SIP authentication on the same SP service is used

Some of the XSI features can be accessed from the phone by launching dedicated apps (such as Network Directories) or via special feature key functions (such as Do Not Disturb) or special soft keys (such as Call Forward All). In addition, all enabled XSI services can also be accessed under the Net Services app of the Main Menu.

Notes:

- CallForwardAll, CallForwardBusy, CallForwardNoAnswer, and DoNotDisturb can also be accessed using the AS-Feature-Sync method. If AS-Feature-Sync is enabled, then it will be used instead of XSI to access these features.
- Except for network directories and call logs, the XSI server will need to notify the phone when settings have changed on the server. This notification is done using a event channel over HTTP. The method for sending HTTP messages is using Comet.

Network Directories

Network Directories are directories hosted by a server somewhere in the network.

BroadSoft Hosted PBX Platform

With BroadSoft's BroadWorks platform, the phone supports the following types of network directories:

- Group
- Group Common
- Enterprise
- Enterprise Common
- Personal

Please consult your BroadSoft documentation on how to setup and manage these directories on the server side.

To access this service from the phone, you must enable the option ***SPn Service – Network Provided Services***::**Directory**. The user can invoke the Network Directory service of a specific SP service from the phone by launching the Net Dir app from the Main Menu of the phone GUI. Which SP service's Network Directories service is invoked is controlled by the parameter ***Phone Settings – Network Directory***::**VoiceService**. In order for the Net Dir item to show on the phone's Main Menu, both ***Phone Settings – Network Directory***::**Enable** and ***SPn Service – Network Provided Services***::**Directory** of the corresponding SP service must be enabled. The user can access the Network Directories on other SP services through the Net Services App.

It should be noted that all the directory data are stored entirely on the server and are downloaded once when the network directories function is invoked on the phone. After initial invocation, user may refresh the data by pressing the "Refresh" or "Refresh All" soft key. There is a "Search" function where user can enter a search string of a name pattern (such as Obi*) and submit it to the server. The server then returns a list of entries matching the search criteria to display on the phone screen.

If Buddy List is also enabled and available under the same SP service, the phone will also display the presence icon in the network directory if it can be found in the buddy list.

Netwrok Directory Soft Key Options

There are 4 soft keys available on the network directory results screen:

- **Call** Call the highlighted entry
- **Search** Enter one or more search fields and press the OK/Enter key to start the search
- **Prev Page** Display the previous page of results

- **Next Page** Display the next page of results

The Search option can be disabled by setting the ***SPn Service – Network Directory Setup***:: **EnableSearch** to `false`. In that case the **Search** soft key is replaced with the **Refresh** soft key which simply reloads the results from the URL.

Other PBX Platforms

A service provider can host a Group directory similar to the format used in BroadWorks on the network. The URL to access this directory is configured in the parameter: ***SPn Service – Network Directory Setup***::**URL**. For example:

URL = https://mypbx.com/group-dir.php?user=jsmith

The returned document should be an XML (properly escaped) as shown in the example below:

```
<?xml version='1.0' encoding='utf-8' ?>
<Group>
  <startIndex>1</startIndex>
  <numberOfRecords>2</numberOfRecords>
  <totalAvailableRecords>2</totalAvailableRecords>
  <groupDirectory>
    <directoryDetails>
      <name>Samuel Samson</name>
      <firstName>Samuel</firstName>
      <lastName>Samson</lastName>
      <userId>ssamson</userId>
      <number>14089991234</number>
      <extension>104</extension>
      <department>Marketing</department>
      <groupId>Management</groupId>
      <impId>ssamson@abcd-wallpaper.com</impId>
      <additionalDetails>
        <emailAddress>sam.samson@abcd-wallpaper.com</emailAddress>
        <mobile>14089998899</mobile>
      </additionalDetails>
    </directoryDetails>
    <directoryDetails>
      <name>John J. Smith</name>
      <firstName>John</firstName>
      <lastName>Smith</lastName>
      <userId>john</userId>
      <number>14087771123</number>
      <extension>103</extension>
      <department>Sales</department>
      <groupId>Staff</groupId>
      <impId>jjs@abcd-wallpaper.com</impId>
      <additionalDetails>
        <emailAddress>john.smith@abcd-wallpaper.com</emailAddress>
        <mobile>16509991802</mobile>
      </additionalDetails>
    </directoryDetails>
  </groupDirectory>
</Group>
```

In addition the following optional URL parameters can be appended to the URL:

- *results* Number of results to return by the server. This is merely a suggestion to the server which may return a different number of results. The phone will not reject the result if the number of results returned is different from the request and will make a best effort to display all the available results

- *start* The 1-based index of the first result to return by the server; the implied start index is 1 if not specified

Example: https://mypbx.com/group-dir.php?user=jsmith&start=30&results=20

You may also include username and password in the URL for authentiation to the user, such as:

https://jsmith:!4xuIKKl2y@mypbx.com/group-dir.php?user=jsmith

Replace the Built-In Phone Book with a Network Directory

You can replace the built-in phone book with a Network Directory, such that when the user presses the **phbk** soft key or selects **Contacts** from the Main Menu, it launches the corresponding network directory instead of the built-in (local) phone book. Below is the configuration for this:

User Preferences – Phone Book Settings::**ActionURL** = `phone://netdir`

Phone Settings – Network Directory::**VoiceService** = `SPn`

(where *n* = 1, 2, 3, ..)

Network Call Logs

The network call logs consist of four logs: All, Missed, Received, and Outgoing. The log data are stored entirely on the server and are downloaded to the phone when the user invoke this function from the phone. Please consult BroadSoft on how to manage these call logs on the server side.

To make this function available on the phone, you must enable the option ***SPn Service – Network Provided Services***::**CallLogs**. There is no specialized app, feature key functions or soft key options to launch network call logs; the user can only invoke this function by going through the Net Services app.

If the Buddy List is also enabled and available under the same SP service, the phone will also display the presence icon in the network directory if it can be found in the buddy list.

BroadWorks Anywhere

The user can view and change the following settings of this feature from the phone GUI:

- Turn on/off the option “Alert all locations for Click-To-Dial Calls”
- Turn on/off the option “Alert all locations for Group Paging Calls”
- Enable/Disable a location
- Add a location
- Remove a location (note: there is a limitation at present where the last location cannot be removed via XSI)
- Edit a location’s Number or Description attributes

To make this function available on the phone, you must enable the option ***SPn Service – Network Provided Services***::**BroadWorksAnywhere**. There is no specialized app, feature key functions or soft key options to launch this function; the user can only invoke this function by going through the Net Services App.

Remote Office

The user can view and change the following settings of this feature from the phone GUI:

- Enable/Disable this feature
- Change the Remote Phone Number – note: there is a limitation at present where the number cannot be removed via XSI.

To make this function available on the phone, you must enable the option ***SPn Service – Network Provided Services***::**RemoteOffice**. There is no specialized app, feature key functions or soft key options to launch this function; the user can only invoke this function by going through the Net Services App.

Simultaneous Ring

The user can view and change the following settings of this feature from the phone GUI:

- Enable/Disable this feature (by turning on/off the "Active" option)
- Turn on/off the option "Do not ring my Simultaneous Ring Numbers if I'm already on a call"
- Add a location
- Remove a location (note: there is a limitation at present where the last location cannot be removed via XSI)
- Change the Phone Number attribute of a location
- Turn on/off the option "Answer confirmation required" for each location

To make this function available on the phone, you must enable the option ***SPn Service – Network Provided Services***::**SimultaneousRing**. There is no specialized app, feature key functions or soft key options to launch this function; the user can only invoke this function by going through the Net Services App.

Call Forward Always

To make this function available on the phone, you must enable the option ***SPn Service – Network Provided Services***::**CallForwardAlways**. This feature is similar to the CallForwardUnconditional feature that is available natively on the phone (per service). To use the version that is provided by the soft-switch, the administrator should disable the corresponding service on the phone to avoid ambiguity.

In addition to going through the Net Services app, the following methods of access are available:

- Feature Key: Define a feature key with the function Call Forward. Set the Service parameter of the key to the SP service that provides this feature
- Soft Key: The Call Forward soft key on the home screen of the phone can be mapped to the network provided CallForwardAll service on a specific SP service, by setting the parameter ***Phone Settings – User Preferences Settings***::**CallForwardUnconditionalFeatureProvider** to equal to that SP service
- User Preferences – Call Forward Setting: As a side effect of enabling the Soft Key option in the last item, the Call Forward Setting under User Preferences app on the GUI is also pointing to the same network provided CallForwardAlways feature

The user can enable/disable CallForwardAlways as well as setting a forwarding number; the settings are stored on the server.

Call Forward Busy

To make this function available on the phone, you must enable the option ***SPn Service – Network Provided Services***::**CallForwardBusy**. This feature is similar to the CallForwardOnBusy feature that is available natively on the phone (per service). To use the version that is provided by the soft-switch, the administrator should disable the corresponding service on the phone to avoid ambiguity.

The only way of access from the phone is by going through the Net Services app. The phone allows the user to get and set this feature on the server; the setting is stored on the server.

Call Forward No Answer

To make this function available on the phone, you must enable the option ***SPn Service – Network Provided Services***::**CallForwardNoAnswer**. This feature is similar to the CallForwardOnNoAnswer feature that is available

natively on the phone (per service). To use the version that is provided by the soft-switch, the administrator should disable the corresponding service on the phone to avoid ambiguity.

The only way of access from the phone is by going through the Net Services app. The user can enable/disable the feature, set the forwarding number, and also the number of rings before forwarding the call; the setting is stored on the server.

Anonymous Call

To make this function available on the phone, you must enable the option ***SPn Service – Network Provided Services***::**AnonymousCall**. This feature is similar to the AnonymousCall feature that is available natively on the phone (per service). To use the version that is provided by the soft-switch, the administrator should disable the corresponding service on the phone to avoid ambiguity.

In addition to going through the Net Services app, the following methods of access are available:

- Feature Key: Define a feature key with the function `Block Caller ID`. Set the Service parameter of the key to the SP service that provides this feature

The user can enable/disable this feature from the GUI; the setting is stored on the server.

Do Not Disturb

To make this function available on the phone, you must enable the option ***SPn Service – Network Provided Services***::**DoNotDistrub**. This feature is similar to the DoNotDisturb feature that is available natively on the phone (per line). To use the version that is provided by the soft-switch, the administrator should disable the corresponding feature on the phone to avoid ambiguity.

In addition to going through the Net Services app, the following methods of access are available:

- Feature Key: Define a feature key with the function `Do Not Disturb`. Set the Service parameter of the key to the SP service that provides this feature

The user can enable/disable this feature from the GUI; the setting is stored on the server.

Built-in Phone Apps

The built-in phone apps can be accessed from the Main Menu. The following apps are available:

Display Caption (Default)	Action URL	Summary
Contacts	Phone://phone-book	A list of contacts where user can add, remove, or modify entries.
Current Calls	Phone://calls	A list of all the current calls on the phone, where user can manipulate the calls individually, or start more calls. The app also pops up automatically when a new call is added
Call History	Phone://call-history Secondary Apps: • Phone://call-history/all • Phone://call-history/received • Phone://call-history/missed • Phone://call-history/outgoing	A list of up to 200 previsous calls with time-stamp, duration, call statistics, and other information for each entry. The call history is further divided into 4 groups: All Calls, Missed Calls, Outgoing Calls, and Received Calls which can be accessed via a corresponding short cut as a secondary app.
Preferences	Phone://preferences	A list of user preferences settings, such as background picture, screen saver, language, screen brightness. The order of the items and contents are configurable
Settings	Phone://settings	A list of phone settings (usually for admin) such as Network, Firmware Update. The order of items and contents are configurable
Product Info	Phone://prod-info	Information about the product such as hardware and software versions, device MAC, etc. This is a read-only list. The order of the items and contents are configurable.
Buddy List	Phone://buddy	A Buddy List service based on XMPP
Net Services	Phone://netsrv	A collection of settings of features that are offered by the service provider. The settings are organized on a per SP Service basis. This App is only applicable with a BroadSoft switch
Net Dir	Phone://netdir	A directory service offered by a service provider. Either BroadSoft directory or LDAP

You can find more information on the operations of these apps in the Obihai IP Phone User Guide. Among these the Net Services and the Net Directory Apps rely on the BroadSoft/BroadWorks switch and are discussed further in the following sections.

Net Services App

BroadSoft offers a set of user features that can be accessed through a few APIs

which corresponds to an SP service on the phone. These features are collected listed under the heading Network Services per SP service on the phone. The settings of these features are made available to the phone user under the Net Services App on the Main Menu.

The top level of the App, a list of the SP services that have at least of the network features enabled shown, along with a short summary of status of some of the enabled features: DND on/off, Call Forward All on/off, ACD Agent State, and Anonymous Call on/off. The soft key set on this screen is configurable with up to 4 options: BCI, DND, CFA, and ACD. User can select to the SP service to view/change all the available settings for the SP service.

The ordering and the list of items to include in the 2nd level menu are configurable (with the NetServicesMenu1 parameter)

Net Dir App

The usage depending on the app provider. There are two options:

1. BroadSoft Directory

2. LDAP

Phone GUI Customization

The OBi1000 has three built-in "Skins" and they dictate primarily the look and feel of the phone GUI. The end user selects a particular skin via the "Preferences" menu on the phone. It is possible to have a custom skin that is downloaded onto the target phone.

A skin is specified at the lowest level with a special (Obihai Proprietary) XML file with its own set of icons. In addition to look and feel, the XML file also dictates, among many other things,

- What menu entries are presented and in what order;
- All the associated wordings;
- What soft keys are being shown and under what circumstances;
- What is shown for the line keys;

Obihai will initially expose the customization options in each aspect to customers at a higher level instead of opening up the XML specification to the public. GUI customization is detailed in a separate document that is available on request from Obihai.

Main Menu

The main menu on the home screen has three optional items that you can show or hide using the following parameters

Phone Settings – Network Services::**Enable** – Enable/Disable the **Net Services** item on the main menu

Phone Settings – Network Directory::**Enable** – Enable/Disable the **Net Dir** item on the main menu

Phone Settings – Buddy List::**Enable** – Enable/Disable the **Buddy List** item on the main menu

The Main Menu can be hidden by default and can be shown or hidden by pressing the HOME Key. To enable this behavior, enable the option ***User Preferences***::**HideMainMenu**.

Line Key Tabs

By default the line keys area to the right of the screen are overloaded with 4 tabs of VLKs such that only one tab is visible at anytime. This feature can be disabled by disabling the option ***User Preferences***::**LineKeyTabs**. When Line Key Tabs are disabled, the phone shows no line key tabs in the status bar and one and only one page of VLKs can be used (VLK 1-6 on OBi1062 and VLK 1-3 on OBi1032); the status icons are also flushed to the right corner. To handle more calls in this case, you may enable multiple calls per Call (Appearance) Key as described in the next section

Controlling Multiple Calls Per Call Key

When defining a Call Key, you can specify the number of calls to control with key using the **MaxCalls** parameter of that feature key. The default value is one; the maximum is four. When there are multiple calls being handled on a Call Key, the VLKW and the LED of the key can only reflect the status of one of the calls. The call whose state is being reflected on the Call Key is referred to as the *focused call* of that key. The focused call is selected by the phone initially based on the call states and the selection is revised when a call state changes, a call is removed from the key, or a new call is added to the key. The user may also force one of the calls to be the focused call by highlighting that call and pressing the "Right" navigation key as described below.

Calls App Behavior

The Calls App shows a list of current calls on the phone. Calls that are controlled under the same Call Key are grouped together in the list, with the calls on Call Keys with a smaller VLK index shown first. New calls are added to the beginning of the group of calls belonging to the same Call Key.

One and only one of the calls on the Calls App screen is highlighted and the corresponding VLKW shows a white bounding box to identify the Call Key that the highlighted call belongs to. As the user navigates through the calls on the screen, the white bounding box will move accordingly. Note that the highlighted call is not necessarily the focused call. If the highlighted call is not the focus, user can press the "Right" navigation key to make it the focused call. Note the action triggered by pressing the Call Key applies to the focused call only.

Normally the calls are packed together on the Calls App screen. Hence it is possible to see two calls belonging to the same Call Key on the same screen. If this is not desirable, you can disable the option ***User Preferences***::**PackCallsOnDisplay** which will cause the phone to show only calls belonging to the same Call Key together on the same screen.

Soft Key Set Customization

The set of soft keys that are available on the phone screen at any given time is called a Soft Key Set. The set is chosen by the phone based on the current App state. Very often it is chosen based on the highlighted item on a screen with multiple items listed. Some of the soft key sets can be customized by entering a list of soft key specifications as a comma-separated list in the corresponding soft key parameters.

Soft Key Set Parameter Syntax

The value of a Soft Key Set parameter is a comma separated list of *soft key specifications*, with the following general format: `{softkey-spec}`[`|{softkey-spec}`], `{softkey-spec}`[`|{softkey-spec}`], `...`

A `{softkey-spec}` is a soft key specification as defined in the next section. The `|{softkey-spec}` syntax lets you specify an optional alternative soft key to show when the given soft key is hidden. A soft key is hidden when its hidden condition is matched. The hidden condition for each soft key is defined internally. For example: `missed|lines` specifies that the **Missed** soft key is shown when it is not hidden, otherwise the **Lines** soft key is shown. If the alternative soft key is an empty key (unspecified or not found), an ***<Empty>*** soft key will be shown as the alternative. Otherwise a hidden soft key is packed and does not occupy any soft key slot on the display.

For example: the set `barge,monitor,,newcall` shows **Barge In**, **Monitor**, ***<Empty>***, **New Call** when barge and monitor are not hidden, and ***<Empty>***, **New Call** when barge and monitor are hidden. On the other hand, the set `barge|,monitor|,,newcall` shows the same when barge and monitor are not hidden, but ***<Empty>***, **<Empty>**, ***<Empty>***, **New Call** when they are.

Soft Key Specification

General format: `{id}[?][;attr[;attr[…]]]`

`{id}` is the standard soft key ID, as shown in the table below.

`?` is an optional syntax to designate if the key should not hidden if the hidden condition is matched.

One or more `attr` attribute elements separated by a semicolon (;) can be included. Attribute is usually in the following format `{attribute-name}[={attribute-value}]`. {attribute-value} may be enclosed in a pair of double quotes (") such that all the enclosed characters are preserved (otherwise all white spaces are removed); the enclosing double quotes are not counted as part of the value. For example: `number=**922222222`, `name="Echo Server"`, or `url="http://abc.com/call.png,auth-token=abcdefg"`. That is, all contents within the double quotes are interpreted as part of the attribute value.

All soft keys support the optional **label** and **icon** attribute which is the text and icon that can be shown on the screen inside the soft key window. Without the label or icon attribute specified the default label and icon of the soft key will be used. Some soft keys that have an on/off (or enabled/disabled or yes/no) state offer a **label1** and **icon1** option that is used when the underlying function is in the "on" state. The **acd** softkey also supports **label2**, **label3**, **icon2**, and **icon3** options for to cover the 4 states of an ACD Agent.

In addition, all soft keys support the optional style attribute that specifies how the soft key with its icons and labels should be styled. The value of the style attribute should be equal to id attribute of one of the <style> elements in the <SoftKeyStyles> XML specifed for the ***Screen Item Customization***::**SoftKeyStyles** parameter. If the style attribute is not specified or not found, the <style> element with id=default will be applied to the soft key, if one is found. Otherwise, the internal default style is used to style the soft key. The following aspects can be specified in a soft key style:

- Soft Key background picture when the key is in normal state, pressed down or highlighted state, and alert state. The picture can be specified as an internal picture file or a http or https URL
- The placement of the label and icon within the soft key window (left/right/center/top/bottom adjust with x and y offset)
- The color, size, and font for the label
- The height and width of the softkey icon and label

Assignable Soft Keys

Below is a list of soft keys that may be assigned to the configurable soft key sets.

Soft Key ID	Description	Hidden Condition	Default Label	Default Label1	Default Label2	Default Label3	Where
aans	(User Preferences) Auto Answer Intercom Calls		Auto Answer	Auto Answer			any
acd	ACD Agent Sign On/Off		Signed Out	Available	Unvailable	Wrapping Up	any
achis	All Calls History		All Calls				any
acturl	Action URL		XML App				any

bac	(User Preferences) Block Anonymous Call
	
	Example:
	`bac;label="Hide CID";`
	`  label1="Show CID";`
	`  number=**9222222222`
			Block Anonymous Call	Block Anonymous Call			any

bci	(User Preferences) Block Caller ID (a.k.a. Anonymous Call)		Block Caller ID	Block Caller ID			any
cfa	(User Preferences) Call Forward Unconditional (a.k.a. Call Forward All) enable/disable.		Call Forward	Call Forwarded			any
chis	Call History		Call History				any
cwa	(User Preferences) Call Waiting enable/disable		Call Waiting				any
dnd	(User Preferences) Do Not Disturb		Do Not Disturb	Do Not Disturb			any
dnr	(User Preferences) Do Not Ring		Do Not Ring	Do Not Ring			any
emlogin	A login utility to collect a username and password from the user and trigger ITSP provisioning (by executing the currently configured ITSP ConfigURL) and use the given username/password for authentication. This step can be used by the ITSP for binding a device to the login username		Prov Login				any
lcr	(Last) Call Return		Call Return				any
ldap	LDAP directory search		LDAP				any
ldn	Last Number Redial		Redial				any
lines	Show a list of available of lines on screen to choose from to make a new		Lines				any

	call						
mchis	Same as missed		Missed				any
missed	Missed Calls History	No New Missed Calls	Missed				any
mwi	Message Status (a.k.a. Message Waiting Indication)		Messages				any
ochis	Same as redial		Redial				any
phbk	Invoke the Phone Book app		Phone Book				any
rchis	Received Calls History		Received Calls				any
redial	Outgoing Calls History		Redial List				any
pg1	Page Group 1		Page Group 1	Page Group 1			any
pg2	Page Group 2		Page Group 2	Page Group 2			any

sd	Speed Dial. It takes four additional optional attributes:
	- **name**: the caller name corresponding to the speed dial number
	- **number**: the speed dial number, which can be full or simple
	- **ptt:** (no value) enable Push-To-Talk option
	- **send:** {digit-codes} to automatically dial some digits after the call, where {digit-codes} is a sequence of the following case-sensitve codes:
	  - 0-9,*,#,a,b,c,d – The DTMF digit to send to the peer. Each digit is sent with 100 ms on and 100 ms off
	  - S – Pause for 3s
	  - s – Pause for 1s
	  - U"{prompt}" – Prompt the user to enter one more digits manually on the phone with the given {prompt} shown on the screen. User then press "OK" soft key to continue
	  - A – Wait for the called party to answer before continuing
	Note that the phones starting executing the first code in {digits} when the call receives early media or when the call is answered, whichever happens first.
	For example:
	`send=Ass1234U"Enter Passcode"5678`
	It is aceeptable to have the send attribute specified in a speed dial without a number (e.g. `sd;send=1234#;label=Park`). In this case the speed dial can
			Speed Dial				any

	be used to send out the list of digits on a connected call, and should only be used in the **CallConnected** soft key set.
	
	Example:
	`sd;label="Echo Test"; number=**9222222222`
	
	`sd;send=1234;label=Park`						

ciurl	Invoke the Call-Info URL of the call as an action URL	Call-Info URL not available	**Call Info**				Ringing, CallConnecting, CallConnected, CallHolding
dial	Call the entered number		**Dial**				Dialing, OnDialing
backspace	Remove the digit or character to the left of the cursor in the input box		**Backspace**				Dialing, OnDialing
mm	Returns to the Main Menu to select an item as the number to call		**Main Menu**				Dialtone
mode	Switching Input Mode		**Switch Mode**				Dialtone, Dialing, OnDialing
switch.line	Switching Line		**Switch Line**				OnDialing
answer	Answer the selected incoming (and ringing) call		**Answer**				Ringing
ignore	Ignore the selected incoming (and ringing) call (via OBiBluetooth); the call will continue to ring on the mobile phone.	Call is not an the OBiBluetooth Service	**Ignore**				Ringing
join	Transfer one conferee to the other in a 3-way (local-mixed) conference call.	Less than or more than two calls on the phone are currently in the Connected state	**Join**				CallConnected
reject	Reject the selected incoming (and ringing) call as busy.		**Reject**				Ringing
newcall	Start a new call with dialtone		**New Call**				Ringing, CallConnected. CallHolding, CallConnecting, CallParked, SCAInUse
end	End the selected call		**End**				CallConnecting CallConnected XferTrying XferRinging XferConnected ConfTrying ConfRinging ConfConnecte

							d
hold	Hold the selected call. This key is mutually exclusive with the Resume key.		**Hold**				CallConnected
privhold	Hold the selected call privately when the call appearance is a SCA.		**Private Hold**				CallConnected
resume	Resume the selected call. This key is mutually exclusive with the Hold key.		**Resume**				CallHolding
conf			**Conference**				CallConnected, CallHolding
add2conf	Resume the holding call to add to the current conference	No other call on the phone that is currently in the Connected state	**Add To Conf**				CallHolding
add2conf	Answer the incoming call to directly add it to the current conference	No other call on the phone that is currently in the Connected state	**Add To Conf**				Ringing

transfer	Transfer the selected call. It takes three additional optional attributes:
	- **name**: the caller name corresponding to the speed dial number
	- **number**: the speed dial number, which can be full or simple
	
	Example:
	`transfer;label="Supervisor";`
	`number=sp2(1034)`
			Transfer				CallConnected, CallHolding

bxfer	Blind transfer the selected call. It takes three additional optional attributes:
	- **name**: the caller name corresponding to the speed dial number
	- **number**: the speed dial number, which can be full or simple
	
	Note that bxfer does not timeout on dialing the target number and does not apply digit map on the dialed number.
	
	Example:
	`bxfer;label="To Vmail";`
	`number=sp1(*28)`
			Blind Transfer				CallConnected, CallHolding

bxfer2	Same as **transfer** softkey w.r.t. collecting the target number from the user, but a blind transfer is done after the target number is collected. In contrast to **bxfer**, **bxfer2** plays dial tone, applies digit map, and would time out as user enters the target number		**Blind Transfer**				CallConnected, CallHolding
cbctl	Invoke conference bridge control app to view who's on the bridge or to apply certain control on the selected	Peer is not a conference	**Bridge**				CallConnected

	participants	bridge	Control				
park	Park the selected call against the current extension		Park				CallConnected, CallHolding
tomobile	Send the call to mobile phone. Only applicable to calls on OBiBluetooth service	Call is not on OBiBluetooth Service	Send To Mobile				CallConnected, CallHolding
dispcode	Enter disposition code for the selected call	Disposition Code service not enabled	Dispose Code				CallConnected, CallHolding
trace	Start a call trace for the selected call	Call Trace service not enabled	Trace				CallConnected, CallHolding
escalate	Escalate the selected call to a supervisor	Escalate service not enabled	Escalate				CallConnected, CallHolding
rec.start	Start call recording	Call Recording service not enabled	Rec.Start				CallConnected, CallHolding
rec.stop	Stop call recording	Call Recording service not enabled	Rec.Stop				CallConnected, CallHolding
rec.pause	Pause call recording	Call Recording service not enabled	Rec.Pause				CallConnected, CallHolding
rec.resume	Resume call recording	Call Recording service not enabled	Rec.Resume				CallConnected, CallHolding
xfer.now	Complete call transfer operation		Transfer Now				XferRinging, XferConnected
conf.now	Start the conference now. That is add the conference target to the conference by resuming also the original call where the conference function is invoked from		Conference Now				ConfRinging, ConfConnected
split	Turn the call to conference target (that has not joined the conference yet) into a regular call, at the same call state. This is equivalent to pressing the Cancel key on the phone while the screen is showing an overlay of the call progess with the conference target		Split				ConfTrying, ConfRinging, ConfConnected
split	Break up a merged conference call item into individual call items on screen with each conferee and places all conferee calls on hold	Highlighted call item is NOT a merged conference call	Split				Conferencing
barge	Barge in a (share) call that is connected or holding at another phone	(Share) Call is not connected or holding	Barge In				SCAInUse
monitor	Monitor a (share) call that is connected at another phone	(Share) Call is not connected	Monitor				SCAInUse
pickup	Pickup a parked call at the highlighted	No call	Pickup				CallParked

	extension	parked					
mute	Selectively mute a call		Mute	UnMute			CallConnected, CallConnecting
coach	Request to coach the call peer, or request to get out of coaching from the call peer		Coach	No Coach			CallConnected
coachee	Enable Coachee mode on this call, or disable Coachee mode on this call		Coachee	No Coachee			CallConnected
nohc	No Hold New Call – Start a new call without placing the current calls on hold first		No Hold Call				CallConnected
divert	Divert a ringing incoming call to a target number that user enters		Divert				Ringing
ns.acd	Set ACD Agent State on the selected SP*n* Service	Feature Sync Disabled	Signed Out	Available	Unavailable	Wrapping Up	NetServices
ns.bci	Turn on/off Anonymous Call feature on the selected SP*n* Service	Feature Sync Disabled	Anonymous	Anonymous			NetServices
ns.dnd	Turn on/off Do Not Disturb feature on the selected SP*n* Service	Feature Sync Disabled	Do Not Disturb	Do Not Disturb			NetServices
ns.exec	Turn on/off Executive Call Filtering feature on the selected SP*n* Service	Feature Sync Disabled	Exec Filter	Exec Filter			NetServices
ns.xass	Turn on/off Executive Assistant Divert on the selected SP*n* Service	Feature Sync Disabled	Exec Assist	Exec Assist			NetServices
ns.rmoff	Turn on/off Remote Office feature on the selected SP*n* Service	Feature Sync Disabled	Rm Office	Rm Office			NetServices
ns.simring	Turn on/off Simultaneous Ringing feature on the selected SP*n* Service	Feature Sync Disabled	Simlt Ring	Simlt Ring			NetServices
ns.cfa	Turn on/off Call Forward All (a.k.a. Unconditional) feature on the selected SP*n* Service	Feature Sync Disabled	Call Forward	Call Forwarded			NetServices
ns.cfb	Turn on/off Call Forward Busy feature on the selected SP*n* Service	Feature Sync Disabled	Cfwd Busy	Cfwd Busy			NetServices
ns.cfna	Turn on/off Call Forward No Answer feature on the selected SP*n* Service	Feature Sync Disabled	Cfwd No Ans	Cfwd No Ans			NetServices
ns.secclass	Get/Set Security Calls on the selected SP*n* Service	Feature Sync Disabled	Sec Class				NetServices
ns.dispcode	Send a call disposition code on the selected SP*n* Service	Feature Sync Disabled	Disp Code				NetServices
ns.hotel	Get/Set Hoteling feature on the selected SP*n* Service	Feature Sync Disabled	Hoteling				NetServices
ns.bwanw	Get/Set BroadWorks Anywhere feature on the selected SP*n* Service	Feature Sync Disabled	BW Anywhere				NetServices
ns.dir	Load/view/refresh the network directory on the selected SP*n* Service	Feature Sync Disabled	Dir				NetServices
ns.clog	Load/view/refresh the network call history on the selected SP*n* Service	Feature Sync Disabled	CLog				NetServices

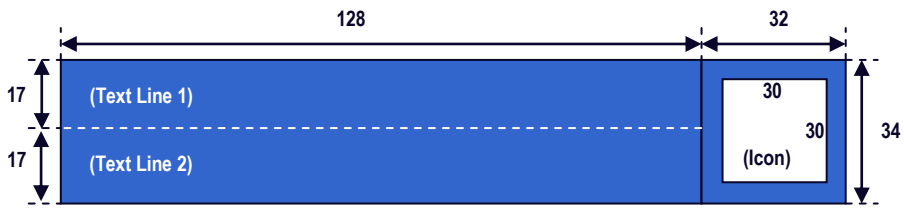
ns.buddy	Load/view/refresh the Buddy List on the selected SP*n* Service	Feature Sync Disabled	**Buddy List**				NetServices
blf.call	Call the monitored extension		**Call**				BLFCall
blf.answer	Answer the highlighted call that is ringing on the monitored extension	**X_BlfDirectedCallPickup** is disabled for the SP*n* Service	**Answer**				BLFCall
blf.pickup	Pickup a call that is currently parked against the monitored extension	**X_BlfCallPickup** is disabled for the SP*n* Service	**Pick Up**				BLFCall
blf.barge	Barge in the highlighted call that is currently active on the monitored extension	**X_BlfBargIn** is disabled for the SP*n* Service	**Barge In**				BLFCall
blf.monitor	Barge in to monitor only the highlighted call that is currently active on the monitored extension	**X_BlfBarge** is disabled for the SP*n* Service	**Monitor**				BLFCall
blf.coach	Barge in to coach (a.k.a. whisper) only the highlighted call that is currently active on the monitored extension	**X_BlfWhisper** is disabled for the SP*n* Service	**Coach**				BLFCall

Soft Key Set Parameters:

Parameter Group	Parameter	Description
IP Phone – Soft Keys – Soft Key Sets	**Home**	The soft key set shown on the Home Screen. The default is `redial,cfa,dnd,missed|lines`
IP Phone – Soft Keys – Soft Key Sets	**SCAInUse**	The soft key set shown on the Calls app screen when the selected call is a shared call appearance being used by another phone. The default is `barge|,monitor|,,newcall`
IP Phone – Soft Keys – Soft Key Sets	**Dialtone**	Dialtone playing, before 1st digit is entered. Default is `redial,phbk,mode,lines`
IP Phone – Soft Keys – Soft Key Sets	**Dialing**	Dialing a number with at least 1 digit entered. Default is `redial,dial,mode,backspace`
IP Phone – Soft Keys – Soft Key Sets	**OnDialing**	On-hook dialing. Default is `switch.line,dial,mode,backspace`
IP Phone – Soft Keys – Soft Key Sets	**CallParked**	On-hook dialing. Default is `switch.line,dial,mode,backspace`
IP Phone – Soft Keys – Soft Key Sets	**CallConnecting**	Trying outgoing call, including the case when called party is ringing. Default is `end,,,newcall`
IP Phone – Soft Keys – Soft Key Sets	**CallConnected**	Call is connected. Default is `end,hold,conf,transfer,privhold,park,dispcode,escalate,trace,rec.start,rec.stop,rec.pause,rec.resume,tomobile`
IP Phone – Soft Keys – Soft Key Sets	**CallHolding**	Call is holding. Default is `end,resume,add2conf,conf,transfer,park,dispcode,escalate,trace,rec.start,rec.stop,rec.pause,rec.resume,tom`

		`obile`
IP Phone – Soft Keys – Soft Key Sets	**Ringing**	Incoming call ringing. Default is `answer,reject,ignore`
IP Phone – Soft Keys – Soft Key Sets	**CallError**	Outgoing call error encountered, such as called party busy or not found. Default is `end,,,newcall`
IP Phone – Soft Keys – Soft Key Sets	**XferTrying**	Trying to call the transfer target. Default is `end`
IP Phone – Soft Keys – Soft Key Sets	**XferRinging**	Transfer target is ringing. Default is `end,,xfer.now`
IP Phone – Soft Keys – Soft Key Sets	**XferConnected**	Connected with the transfer target, including the case when the call is Holding. Default is: `end,hold,resume,xfer.now`
IP Phone – Soft Keys – Soft Key Sets	**ConfTrying**	Trying to call the new conferee `end`
IP Phone – Soft Keys – Soft Key Sets	**ConfRinging**	Conferee is ringing. Default is `end,,conf.now`
IP Phone – Soft Keys – Soft Key Sets	**ConfConnected**	Connected with the conferee, including the case when the call is Holding. Default is `end,hold,resume,conf.now`
IP Phone – Soft Keys – Soft Key Sets	**Conferencing**	A virtual call item representing all the conferee calls in a conference call. All conferee calls by definition are in the Connected state. Default is: `end,split,newcall`
IP Phone – Soft Keys – Soft Key Sets	**NetServices**	The soft keys at the top level of the Network Services app where one of the available SP services is highlighted. Default is: `ns.bci,ns.dnd,ns.cfa,ns.bac,ns.acd`
IP Phone – Soft Keys – Soft Key Sets	**BLFCall**	The soft keys to present when user press+hold a BLF key to bring up a list of calls on the monitored extension. Default is: `blf.call,blf.answer,blf.pickup,blf.barge,blf.coach,blf.monitor`

Line Key Window Customization



A Line Key Window (a.k.a. Screen Tile) is a 160Wx34H-pixel area next to each Line Key on the screen. As shown in the picture above, this window is divided into three non-overlapping regions: Text Line 1, Text Line 2, and Icon. The information shown in the window depends on the function assigned to the corresponding line key (which is a feature key) and the current state of the assigned function.

The information shown on Text Line 1 and 2 can be customized for the following feature key functions: Call Appearance, Speed Dial, Busy Lamp Field, and Action URL, with the following parameters:

Parameter Group	Parameter	Description

Call Appearance		
IP Phone – Feature Key Customization – Call Appearance	**Enable**	Enable the use of customized display for Call Appearance feature keys
IP Phone – Feature Key Customization – Call Appearance	**TextLine1**	Textual information to show on Text Line 1 area during idle
IP Phone – Feature Key Customization – Call Appearance	**TextLine2**	Textual information to show on Text Line 2 area during idle
IP Phone – Feature Key Customization – Call Appearance	**TextLine1InCall**	Textual information to show on Text Line 1 area during a call
IP Phone – Feature Key Customization – Call Appearance	**TextLine2InCall**	Textual information to show on Text Line 2 area during a call
Busy Lamp Field		
IP Phone – Feature Key Customization – Busy Lamp Field	**Enable**	Enable the use of customized display for Busy Lamp Field feature keys
IP Phone – Feature Key Customization – Busy Lamp Field	**TextLine1**	Textual information to show on Text Line 1 area
IP Phone – Feature Key Customization – Busy Lamp Field	**TextLine2**	Textual information to show on Text Line 2 area
IP Phone – Feature Key Customization – Call Appearance	**TextLine1InCall**	Textual information to show on Text Line 1 area when the monitored extension has one or more calls
IP Phone – Feature Key Customization – Call Appearance	**TextLine2InCall**	Textual information to show on Text Line 2 area when the monitored extension has one or more calls
Speed Dial		
IP Phone – Feature Key Customization – Speed Dial	**Enable**	Enable the use of customized display for Spee Dial feature keys
IP Phone – Feature Key Customization – Speed Dial	**TextLine1**	Textual information to show on Text Line 1 area
IP Phone – Feature Key Customization – Speed Dial	**TextLine2**	Textual information to show on Text Line 2 area
Action URL		
IP Phone – Feature Key Customization – Action URL	**Enable**	Enable the use of customized display for Action URL feature keys
IP Phone – Feature Key Customization – Action URL	**TextLine1**	Textual information to show on Text Line 1 area
IP Phone – Feature Key Customization – Action URL	**TextLine2**	Textual information to show on Text Line 2 area
Group *n* (n = 1, 2, 3, 4)		
IP Phone – Feature Key Customization – Group n	**ID**	The ID to reference this Group
IP Phone – Feature Key Customization – Group n	**TextLine1**	Textual information to show on Text Line 1 area
IP Phone – Feature Key Customization – Group n	**TextLine2**	Textual information to show on Text Line 2 area
IP Phone – Feature Key Customization – Group n	**Icon**	Internal icon path or HTTP/HTTPS url

Macros are available to be used in specifying the textual contents for Text Line 1 and Text Line 2 areas. A macro must be preceded with a `$`, such as `$number` or `$(number)`. The enclosing parenthesis is required when the macro is followed by a character that is a legal macro character. A legal macro character is one of {`a-z A-Z . _ 0-9`}.

The following macros are available:

Macro	Description	Notes
$name	The Name field as configured for the feature key	
$number	The Number field as configured for the feature key	
$service	The default name of the service specified in the Service field for the feature key: `SP1`, …, `SP6`, `OBiTALK`, `OBiBluetooth`.	Expands into the word `Auto` if service field is not specified
$service.displayLabel	The X_DisplayLabel parameter per SP*n* service or the DisplayLabel parameter per OBiTALK or OBiBluetooth service	{Blank} if service field is not specified
$service.number	For SP1 – SP6, this is the User ID of the SIP account, which is derived from the AuthUserName or URI parameter of the SP service. For OBiTALK, this is the OBi Number of the phone. For OBiBluetooth, this is the number of the connected cell phone, if available	{Blank} if service field is not specified
$service.displayNumber	The X_DisplayNumber parameter per SP*n* Service or the DisplayNumber parameter per OBiTALK or OBiBluetooth service	{Blank} if service field is not specified
$func	Short internal name of the function assigned to the key: - `call` for Call Apperacne - `blf` for Busy Lamp Field - `spd` for Speed Dial - `acturl` for Action URL	{Blank} if function is not specified
$funcname	Long internal name of the function assigned to the key: - `Call` for Call Apperacne - `Busy Lamp` for Busy Lamp Field - `Speed Dial` for Speed Dial - `XML APP` for Action URL	{Blank} if function is not specified
$call.service and $call.service.*x* where *x* = number, displayLabel, or displayNumber	Same as $service and $service.*x* applying to the service the call is on.	{Blank} if service is undefined for the call (yet)
$call.number	Call peer's Number	
$call.name	Call peer's Name	
$value	For BLF, this is number of calls currently active on the monitored entity. The value is {blank} for other functions	
$blf.ext	The extension of the entity monitored by the feature key	
@id	Line Key Hard Key ID. Exapnds into: - 1,2,3,4,5 for OBi1022 - 1,2,3 for OBi1032/2062/2162 - 1,2,3,4,5,6 for OBi1062/2182	
@p	Line Key Page. Expands into: - 1,2 for OBi1022 - 1,2,3,4 for OBi1032/1062 and OBi2000 Series	
@n	Line Key ID. Expands into: - 1 – 10 for OBi1022 - 1 – 12 for OBi1032/2062/2162 - 1 – 24 for OBi1062/2182	
$eval	See *Using $Macros and @Macros inside XML* section	
@text1	The default information displayed on the (Text Line 1) section of the Line Key Window. Applicable in the TextLine1, TextLine2, TextLine1InCall, and TextLine2InCall paramters	
@text2	The default information displayed on the (Text Line 2) section of the Line Key Window. Applicable in the TextLine1, TextLine2, TextLine1InCall, and TextLine2InCall paramters	
@icon	The default icon displayed on the (Icon) section of the Line Key Window. Applicable to the Icon parameter.	

To further modify the styles to render the contents of a Line Key Window, you can define one or more <styles> in a <LineKeyStyles> XML in the ***Screen Item Customization***::**LineKeyStyles** parameter. With a <style> you can specify the location, font size, color, background image, etc. of the components inside a line key screen tile. The Screen Item Customization parameter group is the subject of the next section.

Example:

In this exercise, we reverse the information that is normally displayed on line 1 and line 2 of all Call Appearance Line Keys. In addition, during a call, we add "--" to the beginning of line 1. Below are the settings for this example:

Parameter Group	Parameter	Value
Call Appearance		
IP Phone – Feature Key Customization – Call Appearance	**Enable**	true (i.e. option checked)
IP Phone – Feature Key Customization – Call Appearance	**TextLine1**	@text2
IP Phone – Feature Key Customization – Call Appearance	**TextLine2**	@text1
IP Phone – Feature Key Customization – Call Appearance	**TextLine1InCall**	-- @text2
IP Phone – Feature Key Customization – Call Appearance	**TextLine2InCall**	@text1

Customizable Screen Elements

The styles of some elements on the phone screen can be customized independently to provide unique look and feel for your deployment. The elements that can be customized are listed in the following table along with the corresponding configuration parameters to specify the customized style. These parameters are under the parameter Group ***Phone Settings – Screen Item Customization***.

Parameter	Description	Example
RingItem	For customizing the Ringing Calls Screen which shows a list of ringing calls. The value must be a <ScreenItem> XML	`<ScreenItem>` `</ScreenItem>`
CallItem	For customizing the Calls Screen which shows a list of all current calls on the phone. The value must be a <ScreenItem> XML	`<ScreenItem>` `</ScreenItem>`

CallTransferItem	For customing the Call Transfer Screen (that shows the call status with the call transfer target). The value must be a valid <ScreenItem> XML	`<ScreenItem>`
		
		`</ScreenItem>`

ScreenSaverCustomContents	For customizing the contents on the screen saver display. The value must be a valid <ScreenItem> XML	`<ScreenItem>`
		
		`</ScreenItem>`

SoftKeyStyles	Contains a list of up to 32 <style> elements to be used for format a soft key. The enclosing root element must be <SoftKeyStyles>. To use a style, specify a style attribute in the custom soft key (in a soft key set parameter), with the value equal to id attribute of the corresponding <style> element. If no style attribute is specified or the given style is not found, the <style> with id="default" is used for the soft key, if one is found. Otherwise the internally defined style is used for the soft key.	`<SoftKeyStyles`
		`  bgimg="http://abc.com/sk1.png"`
		`  hlimg="http://abc.com/sk2.png">`
		`  <style id="default">`
		`  <label align="center"/>`
		`  <icon align="center"/>`
		`  </style>`
		`  <style id= "left">`
		`  <label align="left"/>`
		`  <icon align="right"/>`
		`  </style>`
		`  <style id= "right">`
		`  <label align="right"/>`
		`  <icon align="left"/>`
		`  </style>`
		`</SoftKeyStyles>`

LineKeyStyles	Contains a list of up to 16 <style> elements to be used for format the contents of Line Key. The enclosing root element must be <LineKeyStyles>. To use a style, specify a style attribute in the custom soft key (in a soft key set parameter), with the value equal to id attribute of the corresponding <style> element. If no style attribute is specified or the given style is not found, the <style> with id="default" is used for the soft key, if one is found. Otherwise the internally defined style is used for the Line Key.	`<LineKeyStyles`
		`bgimg="http://abc.com/lkey.png"`
		`>`
		`  <style id="default">`
		`  <text1 align="center"/>`
		`  <text2 align="center"/>`
		`  <icon align="center"/>`
		`  </style>`
		`  <style id= "left">`
		`  <text1 align="left"/>`
		`  <text2 align="left"/>`
		`  <text3 bgimg="" bgcolor="red"`
		`border="1"/>`
		`  <icon align="right"/>`
		`  </style>`
		`</LineKeyStyles>`

TitleBarStyle	This parameter serves to customize the style of the title bar (across the top of the screen). The value must be a valid <TitleBarStyle> XML as shown in the example on the right.
	The height and width
		`<TitleBarStyle>`
		`<time format="@dateFmt"/>`
		`<date format="@timeFmt"/>`
		`<Notifications list="*">`
		`<dnd icon="$eval($state==Enabled?`
		`{http://abc.com/dnd.png}:)"/>`
		`</Notifications>`
		`</TitleBarStyle>`

MainMenu	This parameter servces to customize the main menu of the phone. The value must be a valid	`<MainMenu>`
		`  <item id="phone-book"/>`
		`  <item id="calls"/>`
		`  <item id="call-history"/>`
		`  <item id="sd"`
		`  number="14089991234" name="Ben Thomson"`
		`  />`
		`  <item id="sd"`
		`  number="14089991234" name="Ben Thomson"`
		`  />`
		`</MainMenu>`

MainMenuItem Styles		`<MainMenuItemStyles>`
		`  <style id="default" bgimg="" hlimg=""`
		`  bgcolor="" hlcolor="">`
		`  <icon width="20" height="20"`
		`  xpos="" ypos=""/>`
		`  <icon2 width="20" height="20"`
		`  xpos="" ypos=""/>`
		
		`  </style>`
		`</MainMenuItemStyles>`

In addition to the above, a <ScreenItem> XML can also be used to customize a MenuItem in an OBiPhone XML App: The <ScreenItem> in that case would be defined dynamically in a CDATA block of the corresponding XML App page.

Using $Macros and @Macros inside XML

Using macros in the XML extends the reusability of the XML for different situations. There are two flavors of macros: *variables* with macro names that start with $ and *constants* with macro names that start with @. A $Macro may represent the value of a configuration parameter, an internal state, or other user assigned value. A @Macro may represent a system level constant (that could be different per device model, per GUI Skin, etc.). If the $macro name contains characters that are not (asci or dot (.))

$eval(*expression*)

$Eval is a special macro that evaluates the given expression at run time to produce a result. It must be followed by the expression to evaluate inside parentheses. For example:

- $eval({$pic}==) evaluates to "1" if $pic expands into an empty string, or "0" otherwise.
- $eval($nitems<2) evalutes to "1" if $nitems expands into a string that is numerically less than 2 numerically, or "0" otherwise

This expression is evaluated left to right recursively, starting from the innermost paretheses, to produce a final result that is a simple string without any more operators. The final result can be an empty string. An expression is made up of one, two, or three operands according to the operator. An operand can be the result of a sub-expression which must be enclosed inside a pair of paretheses. Here are some examples:

- $eval(($name==)?Anonymous:$call.name)
- $eval(({$name}==)?{Out Of Area},({$name}==anonymous)?Restricted:{$name})

If an operand contains one or more macros, the macros are fully expanded before the expression is evaluated. The operand after any necessary macro expansion, may contain any characters including white spaces if it is enclosed in braces {}. Otherwise, it can only contain alphanumeric, UTF-8 charaters, `'@'`, `'.'`, `'/'`, and `'_'`. Extra white spaces are allowed in the expression. If the value of a macro cannot be determined, it is a good practice to always enclose operands containing macros with { }.

For instance one could use $eval() in a *display* attribute to show just one of several overlapping elements where the expressions would evaluate to "1" for one and only one of the elements. For example: only one of the following elements, and , will be shown. Similarly for and .

The following table lists the operators that are currently available.

Operator	Description	Examples
==	Syntax: OP1==OP2 Case-insensitve compares OP1 and OP2 and returns "1" if the string is the same, or "0" otherwise	(a==b) returns "0" ((a==b)==0) returns "1"
!=	Syntax: OP1!=OP2 Case-insensitve compares OP1 and OP2 and returns "0" if the string is the same, or "1" otherwise	(3!=0) returns "1"
>>	Syntax: OP1>>OP2 Right arithmetic shift OP1 by OP2 bits if OP2 > 0. Otherwise left arithmetic shift OP1 by (-OP2) bits. OP1 and OP2 are convertd to signed integers before the shift. The result is always a numerical string even if the amount of shift is 0	(3<<0) returns "3" (3<<2) returns "12"
<<	Syntax: OP1>>OP2 Left arithmetic shift OP1 by OP2 bits if OP2 > 0. Otherwise right arithmetic shift OP1 by (-OP2) bits. OP1 and OP2 are converted to signed integers before the shift. The result is always a numerical string even if the amount of shift is 0	(3<=0) returns "0"
>=	Syntax: OP1>=OP2 Compare the numerical value of OP1 and OP2, and returns "1" if OP1 is greater than or equal to OP2, or "0" otherwise. OP1 and OP2 are converted to signed integers before the compare.	(3>=0) returns "1"
<=	Syntax: OP1<=OP2 Compare the numerical value of OP1 and OP2, and returns "1" if OP1 is lessr than or equal to OP2, or "0" otherwise. OP1 and OP2 are converted to signed integers before the compare.	(3<=0) returns "0"
||	Syntax: OP1||OP2 Logical OR the numerical value of OP1 and OP2, and returns "1" if true, or "0" otherwise. OP1 and OP2 are converted to signed integers before the Logical OR.	(3||7) returns "1" (1||0) returns "0"
??	Syntax: OP1??OP2 Return OP1 (unchanged) if it is not an empty string, or OP2 (unchanged) otherwise	(3??2) returns "2" (??me) returns "me"
&&	Syntax: OP1&&OP2 Logical AND the numerical value of OP1 and OP2, and returns "1" if true, or "0" otherwise. OP1 and	(3&&7) returns "1" (3&&8) returns "1"

	OP2 are converted to signed integers before the Logical AND.	(3&&0) returns "0"
>	Syntax: OP1>OP2 Compare the numerical value of OP1 and OP2, and returns "1" if OP1 is greater than OP2, or "0" otherwise. OP1 and OP2 are converted to signed integers before the compare.	(3>7) returns "0"
<	Syntax: OP1<OP2 Compare the numerical value of OP1 and OP2, and returns "1" if OP1 is less than OP2, or "0" otherwise. OP1 and OP2 are converted to signed integers before the compare.	(3<7) returns "1"
+	Syntax: OP1+OP2 Return the sum of the numerical value of OP1 and OP2. OP1 and OP2 are converted to signed integers before the addition.	(3+7) returns "10"
-	Syntax: OP1-OP2 Return the difference of the numerical value of OP1 and OP2. OP1 and OP2 are converted to signed integers before the subtraction.	(3-7) returns "-4"
*	Syntax: OP1*OP2 Return the product of the numerical value of OP1 and OP2. OP1 and OP2 are converted to signed integers before the multiply.	(3*7) returns "21"
&	Syntax: OP1&OP2 Bitwise AND the numerical value of OP1 and OP2. OP1 and OP2 are converted to signed integers before the compare.	(3&7) returns "3"
|	Syntax: OP1|OP2 Bitwise OR the numerical value of OP1 and OP2. OP1 and OP2 are converted to signed integers before the compare.	(3|7) returns "7"
~	Syntax: ~OP1 Bitwise inverse of the numerical value of OP1. OP1 is converted to signed integer before the inversion	(~0x0f) returns "-16"
!	Syntax: !OP1 Logical negation of the numerical value of OP1. OP1 is converted to signed integer before the logical negation	(!321) returns "0" (!0) reurns "1"
%	Syntax: OP1%OP2 Return the remainder of dividing the numerical	(15%3) returns "0" (17%3) returns "2"

	value of OP1 by the numerical value of OP2. OP1 and OP2 are converted to signed integers before the multiply.	
? and :	Syntax: OP1?OP2:OP3 Returns OP2 (unchanged) if the numerical value of OP1 is not "0", or OP2 (unchanged) otherwise	((a==b)?x:y) returns "y"
? and ,	Syntax: OP1?OP2,OP3?OP4,… Return OP2 if the numerical value of OP1 is not "0", otherwise return the result of the next sub-expression after the comma, or return an empty string if there are no more sub-expressions.	(A==B?X,C==D?E) returns an empty-string (A==B?X,C==C?Y) returns Y (A==B?X,D) returns D

Note: >, <, and & must be properly escaped in an XML document.
Note: An integer is 32-bit value from -2147483648 to 2147483647.

Color and Color Pattern

A color value is an integral value where the least significant byte is Blue value (0-255), the next byte the Green value (0-255), and the next byte the Red Value (0-255); the rest of the bytes are not used and must be set to 0. Examples of color value: 0, 31947013, 0xff0000, 0xffff00. In addition, you can use a predefined color name instead of RGB value. The following color names are defined: black, white, red, green, blue, cyan, magenta, yellow, brown, lightred, lightgreen, lightblue, lightcyan, lightmagenta, lightgray, and darkgray.

A color pattern is a comma separated 3-tuple that specifies a starting color, an ending color, and a direction. The phone renders the color by starting the value at the starting color at one bounding edge and gradually varies it until it reaches the ending color at the opposite bounding edge. The direction 0 specifies that the variation is from top to bottom, and 1 from left to right. Examples of color patterns are: "0,0xffffff,1", "red,blue,0", "black,0xffffff,1". If you only specify the starting color, it becomes just a simple color. If you do not specify the direction, the default is 0.

<ScreenItem> XML

This is an XML document with the root element <ScreenItem>. Here is an example of a <ScreenItem> defined for a **CallItem**:

```
<ScreenItem bgcolor="0xcccccc" hlcolor="0xffffff" height="204" height2="102">
  <setvar name="cid" value="http://abc-pub.com/cid-var-tree.php?name=$name&amp;tel=$number"/>

  <!-- A 32-pixel high caption for each item -->
  <span height="32" bgcolor="0x11cccc" font="@gfont-bold" textcolor="0xffffff">
    <img height="30" width="30" xpos="5" ypos="2" src="$stateIcon"/>
    <text align="center" valign="center" size="18">
      <dict ns="cs">$state</dict>   $timer
    </text>
    <text align="right" valign="center" size="18">$index </text>
  </span>

  <!-- Show this if nitems < 2 (fits 1 item on screen) -->
  <span ypos="32" display="$eval($nitems&lt;2)">
    <!-- If a picture path is not given, shown a generic picture:lock.png -->
    <img height="60" width="60" xpos="20" valign="center" src="$pic" display="$eval({$pic}!=)"/>
    <img height="60" width="60" xpos="20" valign="center"
      src="http://192.168.15.225:8080/lock.png" display="$eval($pic==)"/>

    <!-- Show Caller ID: Name, Number, and Organization -->
    <text xpos="110" ypos="50" size="20" width="120" resize="10" wrap="0"
      font="@gfont-bold" textcolor="0">$name</text>
    <text xpos="110" ypos="82" size="18" width="120" resize="10" wrap="0"
      font="@gfont" textcolor="0x050505">$number</text>
    <text xpos="110" ypos="106" size="18" width="120" resize="10" wrap="0"
      font="@gfont" textcolor="0x050505">$cid.org</text>
  </span>

  <!-- Show this if nitems > 1 (fits 2 items on screen) -->
  <span ypos="32" display="$eval($nitems&gt;1)">
```

```
    <!-- If a picture path is not given, shown a generic picture:lock.png -->
    <img height="50" width="50" xpos="20" valign="center"
      src="$pic" display="$eval({$pic}!=)"/>
    <img height="50" width="50" xpos="20" valign="center"
      src="http://192.168.15.225:8080/lock.png" display="$eval({$pic}==)"/>

    <!-- Show Caller ID: Name, Number, and Organization -->
    <text xpos="110" ypos="10" size="18" width="120" resize="10" wrap="0"
      font="@gfont-bold" textcolor="0">$name</text>
    <text xpos="110" ypos="32" size="16" width="120" resize="10" wrap="0"
      font="@gfont" textcolor="0x050505">$number</text>
    <text xpos="110" ypos="52" size="16" width="120" resize="10" wrap="0"
      font="@gfont" textcolor="0x050505">$cid.org</text>
  </span>
</ScreenItem>
```

where for example `http://abc-pub.com/cid-var-tree.php?name=Joe%20Smith&tel=4089991234` would return a "var-tree" XML document such as this:

```
<cid value="1">
  <org value="ABC Publishing, Inc.">
    <title value="VP, Sales"/>
  </org>
  <name value="Joe Smith"/>
</cid>
```

Elements in a <ScreenItem> XML

Element	Attributes	Description
<SreenItem>	Required: height Optional: bgcolor, hlcolor, height2	The root element. The highlighted screen item uses the hlcolor attribute as its background color; other screen items use the bgcolor attribute as their background color. If bgcolor is not specified, the default value is "0xffffff". If hlcolor is not specified, it takes the same value as bgcolor. The height attribute determines the height (in number of pixels) of the item on screen. The maximum value is 204 for OBi1032/1062, 171 for OBi1022, and 360 for OBi2000 Series
	Required: height Optional: align, valign, width, xpos, ypos, bgcolor, textcolor, font, size	defines a rectangular region with its own background color (that overrides the bgcolor/hlcolor attribute of the screen item). It can be used to define for example a caption area for the <ScreenItem>
<img>	Required: src Optional: align, valign, width, height, xpos, ypos, resize	<img> specifies a rectangular region and a URL of a picture to display in that rectangular region
<text>	Required: Optional: align, valign, width, height, xpos, ypos, size, font, textcolor, shadow, hide-text, hidden-if-empty	Tips: Use to insert a white space inside <text>
<dict>	Required: Optional: ns	<dict> may be used inside a <text> element only to translate the enclosed phrase into another one if an entry exists that matches the given text; otherwise the same text is used for the translated text.
<setvar>	Required: name, value	Sets the value of a temporary variable with the given name within the scope of the <ScreenItem> it is defined in. Normally you would use ${name} in the same <ScreenItem> to use the value of the defined variable (any undefined variable would evaluate to an empty string). A special case is when the value attribute evaluates to an http/https URL, in which case

		the named variable refers to a tree of variables, called a *var-tree*, defined in the XML document downloaded from the given URL.
		
		For example: <setvar name="test" value="my name"/> sets the variable $test to the string "my name". But in <setvar name="cid" value="http://abc.com/cid-strings.xml"/> $cid represents a tree of variables derived from the structure of the document cid-strings.xml. A valid var-tree XML must have the underlying variable name as the root element. For the last example, it would look something like this:
		
		`<cid value="1">`
		`  <org value="ABCD Publishing, Inc.">`
		`  <title value="VP, Sales"/>`
		`  </org>`
		`  <name value="Joe Smith"/>`
		`  <pic`
		`  value="http://abcd.com/pics/small/joe.smith.png"/>`
		`</cid>`
		
		A child variable name at each level is formed by concatenating the XML tag-names of all the ancestors with a '.'. The variables and corresponding (string) values defined in the above XML are therefore:
		$cid = 1
		$cid.org = ABCD Publishing, Inc.
		$cid.org.title = VP, Sales
		$cid.name = Joe Smith
		$cid.pic = http://abcd.com/pics/small/joe.smith.png
		
		Note that the *value* attributes in a var-tree are optional. Note also that giving the root element a simple value "1" would make it easy to test in the <ScreenItem> if the $cid object is defined or not, as can be used for example in the following block:
		`<span display="$cid"> … </span>`
		
		When it is required to have the value interpreted literally and not as a http URL to fetch a var tree, use the value2 attribute instead of the value attribute.

<time>	Required: format Optional: align, valign, width, height, xpos, ypos, size, font, textcolor, shadow	Renders a time string of the current local time in the given format
<date>	Required: format Optional: align, valign, width, height, xpos, ypos, size, font, textcolor, shadow	Renders a date string of the current local date in the given format

Attributes in a <ScreenItem> XML

Attribute	Elements	Description
height	<ScreenItem>, , <text>, <img>	Height of the item's bounding rectangle in pixels
height2	<ScreenItem>	The height of the item's bounding rectangle in pixels, when the number of items is greater than 1. If the value is not specified, the value of the "height" attribute is used for any number of items.
width	, <text>, <img>	Width of the bounding rectangle in pixels
xpos	, <text>, <img>	x-offset from the upper left corner of the parent bounding rectangle in pixels
ypos	, <text>, <img>	y-offset from the upper left corner of the parent bounding rectangle in

		pixels
bgcolor	<ScreenItem>,	Background color or color pattern of the element. Examples: "0xaaaaaa,0xdddddd,1", "white", "black,0xff0000,0"
hlcolor	<ScreenItem>	The background color or color pattern to use when the item is highlighted. If not specified, it is taken as the same as the bgcolor attribute.
bgimg	<ScreenItem>,	Background image to fill the bounding rectangle. The value can be the full path of a locally stored picture, or a http/https URL. If this value is specified, bgcolor and hlcolor are ignored.
hlimg	<ScreenItem>	Background image to fill the bounding rectangle when the <ScreenItem> is highlighted. The value can be the full path of a locally stored picture, or a http/https URL. If the value is not specified, the value of bgimg is used. If either hlimg or bgimg is specified, hlcolor is ignored.
textcolor	, <text>	Foreground color to render the display text
font	, <text>	The font to use to render the display text. The only allowed values are: @gfont and @gfont-bold
size	, <text>	The font size (in pixel height) to use, such as: 12, 14, 18
wrap	<text>	The value should evaluate to “0” or “1”, to determine if the text should auto-wrap to the next line when running out of space on the current line.
resize	<text>, <img>	The value should evaluate to “0” or “1”, to determine if the text should auto-shrink to to fit on the current line. If the text does not fit and neither wrap or resize is enabled, the text will be truncated
align	<text>, <img>	Specifies the horizontal alignment of the widget within the parent rectangle. Valid values are: “left”, “right”, or “center”. Default is “left”
valign	<text>, <img>	Specifies the vertical alignment of the widget within the parent rectangle. Valid values are: “top”, “bottom”, or “center”. Default is “top”
ns	<dict>	Specifies the namespace to search for a dictionary entry for the given phrase
src	<img>	The value should be either an internal picture file or an http/https URL of an image file, such as http://abc.com/cid_pics/user-pic?number=$number
display	, <text>, <img>	Value should be evaluated to a “0” to hide the element, or “1” to show the elment on the screen
format	<time>, <date>	The specific format to present the value of the widget. For <time>, the format string is made up of one or more of the following (case-sensitive) codes: - H = hour shown in 24-hr format in 1 or 2 digits, such as 3 or 16 - H2 = hour shown in 24-hr format in 2 digits, such as 03 - h = hour shown in 12-hr format in 1 or 2 digits, such as 3 or 11 - h2 = hour shown in 12-hr format in 2 digits, such as 03 - m = minute shown in 1 or digits, such as 3 or 24 - m2 = minute shown in 2 digits, such as 03 - s = second shown in 1 or 2 digits, such as 5 or 33 - s2 = second shown in 2 digits, such as 05 - apm = show am or pm in lower case - APM = show AM or AM in upper case - Other characters, such as ‘:’ are treated as separator - Examples: "h2:m2:s2 apm","H:m2:s2" For <date>, the fomat is made up of one or more of the following (case-sensitive) codes: - w = Weekday in full, such as Tuesday - w3 = Weekday in 3 letters, such as Tue - m = Month in full, such as March - m3 = Month in 3 letter, such as Mar - mn = Month in number, such as 3 - d = Day in number (1 – 31) - y = Year in full number, such as 2015 - y2 = Year in 2 digits, such as 15 - Other characters such as ‘/’ are treated as separator

		- Examples: "w mn/d/y", "w3 m d, y"
shadow	<text>, <time>,<date>	Three comma-separeted values "{x-offset },{y-offset },{color}" that specify a shadow to be drawn for the text to be displayed. {x-offset} and {y-offset} are, respectively, the horizontal and vertical offset in pixels, and {color} is the RGB color of the shadow. For example: “2,1,0x444444”
hide-text	<text>	Valid values are '0' (default) and '1'. A value of '1' makes it not to render the text (but could still render the bgimg and bgcolor if specified in the element)
hidden-if-empty	<text>	Valid values are '0' (default) and '1'. A value of '1' hides the widget completely when the text to render is an empty string (such that any bgimg or bgcolor will not be shown either if specifid in the element)

Macors that can be used inside a <ScreenItem>

Macro	Where	Description
$nitems	Anywhere inside a <ScreenItem>	Number of items currently shown on the screen
$name	Anywhere inside a <ScreenItem>; usually used inside a <text> element or src attribute	For a CallItem or RingItem, this expands into the peer’s Caller-ID, or an empty string if Caller-ID Name is not available. For other types of <ScreenItem>, this expands into an empty string.
$number	Anywhere inside a <ScreenItem>; usually used inside a <text> element or src attribute	For a CallItem or RingItem, this expands into the peer’s Caller-ID Number, or an empty string if the Caller-ID Number is not available. For other types of <ScreenItem>, this expands into an empty string.
$pic	Anywhere inside a <ScreenItem>; usually used inside a src attribute	For a CallItem or RingItem, this expands into the path/URL to get the Caller-ID Picture, or an empty string if Caller-ID Picture is not available. Calller-ID picture information is extracted from SIP signaling messages. For other types of <ScreenItem>, this expands into an empty string.
$org	Anywhere inside a <ScreenItem>; usually inside a <text> element	For a CallItem or RingItem, this expands into the peer’s organization information that is extracted from SIP signaling messages, or an empty string if organization information is not available. For other types of <ScreeItem>, this expands into an empty string.
$timer	Anywhere inside a <ScreenItem>; usually inside a <text> element	For a CallItem or RingItem, this expands into the call timer string that counts the duration of the call, that is displayed in h:m:s format on the screen, at 1s resolution. For other types of <ScreenItem>, this expands into an empty string.

$state	Anywhere inside a <ScreenItem>; usually inside a <text> or <dict> element	For a CallItem or RingItem, this expands into the current call state, which can be one of the following values:
		`Trying`
		`Peer Ringing`
		`Ringing`
		`Connected`
		`Holding`
		`Call Ended`
		`Incoming Page`
		`Line Seize`
		`SCA In Use`
		`Call Parked`
		
		For other types of <ScreenItem>, this expands into an empty string.

$stateIcon	Anywhere inside a <ScreenItem>; usually inside a src attribute	For a CallItem or RingItem, this expands into the internal icon file that represents the current call state. For other types of <ScreenItem>, this expands into an empty string.
$index	Anywhere inside a <ScreenItem>; usually inside a <text> element	For a CallItem or RingItem, this expands into the index information of the item, such as 1/2, 2/2, etc. This is only applicable if the call item is grouped under a Line Key that controls multiple call appearances. It is an empty

		string if the underlying Line Key controls just one call appearance.
$coachIcon	Anywhere inside a <ScreenItem>, usually inside a src attribute	For a CallItem or RingItem, this expands into an internal icon that indicates if the current call is in Coaching mode or Coachee mode. The icon is hidden if the call is in neither mode.
$coachState	Anywhere inside a <ScreenItem>	For a CallItem, this expands into coach or coachee if the call is either in coaching or coachee mode. Otherwise it is an empty string.
$charges	Anywhere inside a <ScreenItem>	For a CallItem, when "Advice Of Charges" information is available, this expands into a string that reflects the current charges for the call (such as $ 1.23). The string is automatically updated automatically every few seconds (as soon as the server sends an update).
$muted	Any child element inside <ScreenItem>	For a CallItem, this expands into either 1 if the call leg is selectively muted (with the **MUTE** softkey) or 0 otherwise
$recst	Any child element inside <ScreenItem>	For a CallItem, this expands into either on, off, or paused to reflect the current call recording state. If call recording service is not available at all, it is an empty string
$eval	Any child element inside <ScreenItem>	See section heading **$eval()**
$prompt	Any child element inside <ScreenItem>. Usually used inside a <text> element or src attribute	For a <MenuItem> in an OBiPhone XML App, this expands into the value of the <Prompt> element inside the <MenuItem>.
$url	Any child element inside a <ScreenItem>. Usually inside a <text> element or src attribute	For a <MenuItem> in an OBiPhone XML App, this expands into the value of the <Dial> element if it is present, or the <URL> or <URI> element inside the <MenuItem>.
$icon	Any child element inside a <ScreenItem>. Usually inside a src attribute	For a <MenuItem> in an OBiPhone XML App, this expands into the icon (in an IconList) referenced in the icon attribute of the <MenuItem>.
$f*n* *n* = 1 – 4	Any child element inside a <ScreenItem>. Usually inside a <text> element or src attribute.	For a <MenuItem> in an OBiPhone XML App, this expands into the value of the corresponding <F*n*> element inside the <MeniItem>.
@gfont	Inside a font attribute	The system default text font
@dateFmt	Inside the format attribute of a <date> element	The system default date format
@timeFmt	Inside the format attribute of a <time> element	The system default time format

Additional macros that can be used in a CallItem when it is a (merged) conference call item:

Macro	Where	Description
$conf	Anywhere inside a <ScreenItem>	Expands into number of conferees in a merged conference call item; otherwise the value is 0. That is, the value is 2 for a 3-way conference and 3 for a 4-way conference
$name2 $name3	Anywhere inside a <ScreenItem>; usually used inside a <text> element or src attribute	Whereas $name is a property of the first conferee, this is the corresponding property for the second and third conferee respectively
$number2 $number3	Anywhere inside a <ScreenItem>; usually used inside a <text> element or src attribute	Whereas $number is a property of the first conferee, this is the corresponding property for the second and third conferee respectively
$pic2 $pic2	Anywhere inside a <ScreenItem>; usually used inside a src	Whereas $pic is a property of the first conferee, this is the corresponding property for the second and third conferee respectively

	attribute	
$org2 $org2	Anywhere inside a <ScreenItem>; usually inside a <text> element	Whereas $org is a property of the first conferee, this is the corresponding property for the second and third conferee respectively
$timer2 $timer3	Anywhere inside a <ScreenItem>; usually inside a <text> element	Whereas $timer is a property of the first conferee, this is the corresponding property for the second and third conferee respectively
$charges2 $charges3	Anywhere inside a <ScreenItem>	Whereas $charges is a property of the first conferee, this is the corresponding property for the second and third conferee respectively
$muted2 $muted3	Any child element inside <ScreenItem>	Whereas $muted is a property of the first conferee, this is the corresponding property for the second and third conferee respectively

<LineKeyStyles> XML

This is an XML document that specifies up to 16 styles that a Line Key can be displayed with. Recall that a Line Key screen tile has 3 components that can be styled: TextLine1, TextLine2, and Icon. If the Line Key has the function field set to `Call Appearance`, a 4th element <icon2> is also available. In addition, the width, height, xpos, ypos, background color and background image, and the border around the screen tile can also be styled.

The root element is <LineKeyStyles> which encloses up to 16 <style> element as shown in the below skeleton:

```
<LineKeyStyles ...>
  <style id="default" ...>
    <text1 .../>    <!-- TextLine1 Style -->
    <text2 .../>    <!-- TextLine2 Style -->
    <text3 .../>    <!-- Small Text ovelays to show number of calls on the key -->
    <icon .../>     <!-- Icon Style -->
  </style>
  ...
  <style id="xyz" ...>
    <text1 .../>
    <text2 .../>
    <icon .../>
    <icon2 .../>
  </style>
<LineKeyStyles>
```

A Line Key Window is a 160Wx34H-pixel area (120Wx34H on the OBi1022 and 220Wx60H on the OBi2000 Series) evenly spaced on the right of the screen. This window defines the root bounding box of each soft key. With <LineKeyStyles> you can define an area that is equal to or smaller than the root bounding box and place it anywhere within the root bounding box (using the xpos, ypos, width and height attributes for the <LineKeyStyles> or <style> elements). The attributes defined in <LineKeyStyles> are inherited by all the child <style> elements; each <style> element can define its own set of attributes to overwrite what are inherited).

Elements in a <LineKeyStyles> XML

Element	Attributes	Description
<LineKeyStyles>	Required: Optional: bgimg, hlimg, width, height, xpos, ypos, font, size, textcolor, border, bordercolor	Root element that contains all the <style> elements. The attributes define the default attribute values to use in each <style> element if they are not specified in <style>.
<style>	Required: id Optional: bgimg, hlimg, width, height, xpos, ypos, font, size, textcolor, border, bordercolor, hidden	Each <style> defines a style that can be referenced by id by a Line Key in the Style field of that Line Key parameter. A <style> may be referenced by 0 to many Line Keys. By default the Style field of all Line Key parameters is blank to refer the <style> with the id "default". If an attribute is not specified in <style>, it inherits the value from the same attribute in <LineKeyStyles>

<text1>	Required: Optional: width, height, xpos, ypos, font, size, textcolor, align, valign, hidden	This optional child element of <style> specifies the style to render the TextLine1 component of the referencing Line Key screen tile.
<text2>	Required: Optional: width, height, xpos, ypos, align, valign, hidden	This optional child element of <style> specifies the style to render the TextLine2 component of the referencing Line Key screen tile.
<icon>	Required: Optional: width, height, xpos, ypos, align, valign, hidden	This optional child element of <style> specifies the style to render the Icon component of the referencing Line Key screen tile
<text3>	Required: Optional: width, height, xpos, ypos, font, size, align, textcolor, align, hidden, bgimg, bgolor, border	This optional child element of <style> specifies the style to render the supplementary text overlay on the referencing Line Key screen tile. This element only applies if the function assigned to the Line Key is "Call Appearance" where the supplementary text shown is the number of active calls on the key

Attributes in a <LineKeyStyles> XML

Attribute	Elements	Description
xpos	<LineKeyStyles>, <style>, <label>, <icon>	x position of the upper left corner pixel of the bounding box within the parent bounding box
ypos	<LineKeyStyles>, <style>, <label>, <icon>	y position of the upper left corner pixel of the bounding box within the parent bounding box
align	<label>, <icon>, <icon2>	Horizontal alignment of the widget within its own bounding box. Note that the bounding box's own location within its parent bounding box is determined by the xpos and ypos attributes
valign	<label>, <icon>, <icon2>	Vertical alignment of the widget within its own bounding box
width	<LineKeyStyles>, <style>, <label>, <icon>, <icon2>	Width of the bounding box
height	<LineKeyStyles>, <style>, <label>, <icon>, <icon2>	Height of the bounding box
bgcolor	<LineKeyStyles>, <style>	Background color or color pattern for the Line Key screen tile. Note that if bgimg is specified and available, it will cover the background color. Example: <style bgimg="" bgcolor="white,blue,0" border="2">> Note that we set bgimg to empty string to overwrite the default bgimg; otherwise the bgcolor will be covered by the default bgimg.
bgimg	<LineKeyStyles>, <style>	Background image. It can be an internal path or a http/https URL
id	<style>	The ID to refer to this style in the Style field of a Line Key parameter
font	<LineKeyStyles>, <style>, <label>	Text font (the value can only be "@gfont" or "@gfont-bold")
size	<LineKeyStyles>, <style>, <label>	Text size
textcolor	<LineKeyStyles>,	Text color

	<style>, <label>	
border	<LineKeyStyles>, <style>	The default is "0" means no border around the bounding box. "*n*" means a border of *n*-pixel around the bounding box
bordercolor	<LineKeyStyles>, <style>	The color of the border to draw around the bounding box of the widget
src	<icon>, <icon2>	Overwrites the icon's file path or URL

Macors that can be used in a <LineKeyStyles> XML

Macro	Where	Description
$eval	Anywhere	See section heading **$eval()**
@id	Anywhere	Line Key Hard Key ID. Exapnds into: - 1,2,3,4,5 for OBi1022 - 1,2,3 for OBi1032/2062/2162 - 1,2,3,4,5,6 for OBi1062/2182
@p	Anywhere	Line Key Page. Expands into: - 1,2 for OBi1022 - 1,2,3,4 for OBi1032/1062 and OBi2000 Series
@n	Anywhere	Line Key ID. Expands into: - 1 – 10 for OBi1022 - 1 – 12 for OBi1032/2062/2162 - 1 – 24 for OBi1062/2182

Example 1: Double Size Line Key Screen Tiles

In this example, the defaut line key <style> doubles the normal size of the Line Key Tiles which is suitable for an OBi1032 which has just three Hard Line Keys. Below are the parameter settings and a screen shot of the result.

Line Keys – Key 1::**Function** = `Call Appearance`

Line Keys – Key 2::**Function** = `Call Appearance`

Line Keys – Key 3::**Function** = `Call Appearance`

Line Key Customization – Call Appearance::**Enable** = `true` (checked)

Line Key Customization – Call Appearane::**TextLine1** = `@id:@p:@n`

Line Key Customization – Call Appearane::**TextLine2** = (blank)

Screen Item Customization::**LineKeyStyles** =

```
<LineKeyStyles ypos='$eval((((@n-1)%3)*68)+2)' height='64'>
  <style id='default'>
    <text1 xpos='0' align='left'/>
  </style>
</LineKeyStyles>
```

<SoftKeyStyles> XML

This is an XML document that specifies up to 32 styles that a custom soft key can be displayed with. Recall that a soft key has 2 components that can be styled:

- Label: The label (text) for the soft key
- Icon: The icon for the soft key

In addition, the width, height, xpos, ypos, background color and image, and the border around the screen tile can also b styled.

The root element is <SoftKeyStyles> which encloses up to 32 <style> element as shown in the below skeleton:

```
<SoftKeyStyles ...>
  <style id="default" ...>
    <label .../>
    <icon .../>
  </style>
  ...
  <style id="xyz" ...>
    <label .../>
    <icon .../>
  </style>
<SoftKeyStyles>
```

A Soft Key Window is a 120Wx34H-pixel area (80Wx34H on the OBi1022 and 200Wx60H on the OBi2000 series) evenly spaced on the bottom of the screen. This window defines the root bounding box of each soft key. With <SoftKeyStyles> you can define an area that is equal to or smaller than the root bounding box and place it anywhere within the root bounding box (using the xpos, ypos, width and height attributes for the <SoftKeyStyles> or <style> elements). The attributes defined in <SoftKeyStyles> are inherited by all the child <style> elements; each <style> element can define its own set of attributes to overwrite the default values that are inherited.

Elements in a <SoftKeyStyles> XML

Elements	Attributes	Description
<SoftKeyStyles>	Optional: bgimg, hlimg, width, height, xpos, ypos, font, size, textcolor, border, bordercolor	Root element that contains all the <style> elements. The attributes define the default attribute values to use in each <style> element if they are not specified in the <style>.
<style>	Required: id; Optional: bgimg, hlimg, width, height, xpos, ypos, font, size, textcolor, border,	Each <style> defines a style that can be referenced by id by a Soft Key definition (via the style attribute) in a Soft Key Set parameter. A <style> may be referenced by 0 to many Soft Keys. By default all Soft Keys reference the <style> with the id "default" if the style attribute is not specified in the soft key definition.

	bordercolor, hidden	If an attribute is not specified in <style>, it inherits the value from the same attribute in <SoftKeyStyles>
<label>	Optional: width, height, xpos, ypos, font, size, textcolor, align, valign, hidden	This optional child element of <style> specifies the style to render the label (text) of the soft key. It inherts the following attributes from its ancestors: font, size, textcolor
<icon>	Optional: width, height, xpos, ypos, font, size, textcolor, align, valign, hidden,src	This optional child element of <style> specifies the style to render the icon of the soft key.

Attributes in a <SoftKeyStyles> XML

Attribute	Elements	Description
xpos	<SoftKeyStyles>, <style>, <label>, <icon>	x position of the upper left corner pixel of the bounding box within the parent bounding box
ypos	<SoftKeyStyles>, <style>, <label>, <icon>	y position of the upper left corner pixel of the bounding box within the parent bounding box
align	<label>, <icon>	Horizontal alignment of the widget within its own bounding box. Note that the bounding box's own location within its parent bounding box is determined by the xpos and ypos attributes
valign	<label>, <icon>	Vertical alignment of the widget within its own bounding box
width	<SoftKeyStyles>, <style>, <label>, <icon>	Width of the bounding box
height	<SoftKeyStyles>, <style>, <label>, <icon>	Height of the bounding box
bgcolor	<SoftKeyStyles>, <style>	Background color or color pattern. Note that if bgimg is specified and available, it will cover the background color. Example: `<style bgimg="" hlimg="" hlcolor="black" bgcolor="$eval($alert?{red,black,1}:{white,black,1})">` Note that in the example we set bgimg and hlimg to empty string to remove overwrite the default bgimg or hlimg, otherwise the bgcolor and hlcolor will be covered.
hlcolor	<SoftKeyStyles>, <style>	Background color or color pattern when the key is highlighted or pressed. Note that if hlimg is specified and available, it will cover the background color when the key is hightlighted or pressed.
bgimg	<SoftKeyStyles>, <style>	Background image. It can be an internal path or a http/https URL
hlimg	<SoftKeyStyles>, <style>	Background image to show when the soft key is highlighted or pressed. It can be an internal path or http/https URL
id	<style>	The ID to refer to this style in a custom soft key configuration
font	<SoftKeyStyles>, <style>, <label>	Text font (the value must be either "@gfont" or "@gfont-bold")
size	<SoftKeyStyles>, <style>, <label>	Text size.
textcolor	<SoftKeyStyles>, <style>, <label>	Text color
border	<SoftKeyStyles>, <style>	The default is "0" means no border around the bounding box. "n" means a border of n-pixel around the bounding box
bordercolor	<SoftKeyStyles>, <style>	The color of the border to draw around the bounding box of the widget
src	<icon>	Overwrites the soft key icon's file path or URL

Macors that can be used in a <SoftKeyStyles> XML

Macro	Where	Description
$alert	Anywhere. Typically inside an $eval expression in an attribute	$alert expands into "1" for the following soft keys: - **cfa** if Call Forward All/Unconditional is enabled - **mwi** when there are new messages - **missed** when there are new missed calls - **dnd** when Do Not Disturb is enabled - **dnr** when Do Not Ring is enabled - **cwa** when Call Waiting is disabled - **aans** when Auto Answer Intercom is disabled - **bci** when Block Caller ID is enabled - **bac** when Block Anonymous Call is enabled - **pg1,pg2** when the corresponding group is joined Otherwise, it expands into "0". For example: `bgimg="$eval($alert?{http://xyz.com/a.png}:{http://xyz.com/b.png})"`
$eval	Anywhere	See section heading **$eval()**

Example 1: Use Icons Only for Home Screen Soft Keys

In this example, we will make all the soft keys on the Home Screen to show only an icon that is center justified. The icons we use are all internal icon files, that you may replace with external http/https URLs. Here are the parameter settings and a screen shot of the result.

***Soft Key Sets*::Home** =

```
redial;icon=dial.png;style=1,cfa;icon=cfwd.png;icon1=cfwd.png;style=1,dnd;
icon=dnd_tb.png;icon2=dnd_tb.png;style=1,lines;icon=conf.png;style="1"
```

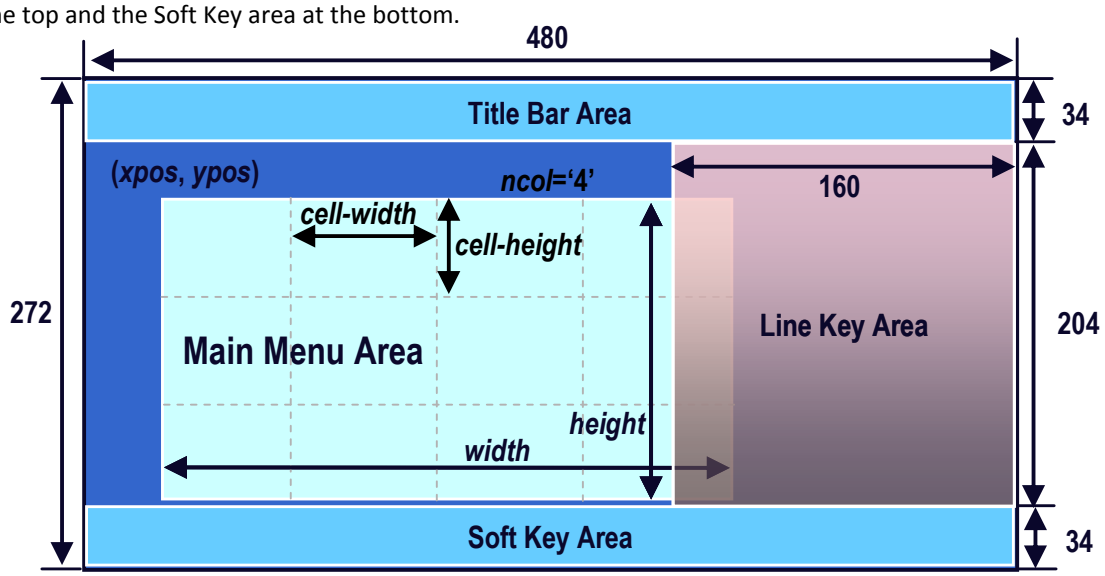
***Screen Item Customization*::SoftKeyStyles** =

```
<SoftKeyStyles>
  <style id='1'>
    <label hidden='1'/>
    <icon width='120' align='center'/>
  </style>
</SoftKeyStyles>
```

Main Menu Customization

The Main Menu is the first menu that appears on the phone screen when the phone starts up, unless it is hidden by default by configuration, in which case the user would press the 'Home' key to show/hide the Main Menu. In order to use a customized Main Menu, you must enable the option ***User Preferences***::**CustomMainMenu** and also define a <MainMenu> XML in the parameter ***Screen Item***::**MainMenu**.

As shown in the picture below (for OBi1062/1032), the Main Menu area occupies the screen area between the Title Bar area at the top and the Soft Key area at the bottom.



The Main Menu can occupy the full width of the screen when the Line Key area is hidden. When the Line Key area is visible, however, it is shown on top of the Main Menu if they overlap. In the <MainMenu> XML, you can use the macro $sys.lines to check if the Line Keys is visible ("1" means visible and "0" means not). You may specify the position and dimension of the Main Menu with the usual four attributes of the <MainMenu> element: xpos, ypos, width, and height. The pair (xpos, ypos) specifies the upper left corner of the Main Menu; the value (0, 0) coincides with the lower left corner of the Title Bar Area.

The Main Menu is made up of a 2-dimensional array of items with a fixed number of columns based on configuration and a number of rows based on the total number of visible items in the menu. When the number of rows exceeds what the screen can show at one time, user can scroll up/down to view the other pages of menu items. Each item occupies a small rectangular area with the same pixel dimension that is represented by the cell-width and cell-height attributes of

the <MainMenu> element. The following is an exmple of a customized Main Menu (showing page 1 of 2) with Line Keys hidden, ncol="6", width="480", height="204", cell-wdith="80", cell-height="68", scrollstyle="2".

The full customization of the Main Menu is divided into two layers. The first layer defines the contents of the Main Menu, such as the dimension of 2-D array, the number of items in the menu and what each item is and does. These are specified with a <MainMenu> XML in the ***Screen Item Customization***::**MainMenu** parameter. The second layer defines the detail styling of each main menu items or a group of main menu items. These are specified with a <MainMenuItemStyles> XML in the ***Screen Item Customization***::**MainMenuItemStyles** parameter. These XML are the subject of the next two sections

<MainMenu> XML

General Structure

```
<MainMenu ...>
  <item id="x" ... />
  ...
  <item id='y' ... />
</MainMenu>
```

The only required attribute of a <MainMenu> <item> element is *id*, which specifies the type of the item. All other attributes are optional with well-defined default values if not specified. Each item is made up of the following components:

- background image: a different image can be specified for when the item is highlighted and not highlighted via the hlimg and bgimg attributes respectively
- background color: the fill color of the item's background when background image is not specified or not the image does not conver the entire background. A different background color can be specified for when the item is highlighted and not highlighted via the bgcolor and hlcolor attributes
- label: a text string by default shown on the bottom of item, usually used as the title of item
- label2: an additional general purpose text string
- icon: an icon by default shown at the center of the item, usually to reflect the current state of item. You may specify an internal path of the picture file or a http/https URL via the icon attribute
- icon2: an additional general purpose icon. You may specify an internal path of the picture file to use, or a http/https URL via the icon2 attribute

To finely control the various aspects of each component (such as position, size, font, color, etc.), you can specify a *style* attribute to select a style to use for the item. The value of *style* must be the *id* of a <style> that is configured in the ***Screen Item Customization***::**MainMenuItemStyles** parameter. If *style* attribute is not specified, the <style> with id = "default", if present, is used to style the item.

Elements in <MainMenu>

Element	Attributes	Description
<MainMenu>	Required: Optional: ncol, width, noscroll, scrollstyle, c1, c2, c3, c4, cell-width, cell-height, hlborder, hlbordercolor, textcolor, hltextcolor, font, size	Root element.
<item>	Required: Optional: id, label, number, name, service, refresh-id	One of more allowed. Each <item> describes one tile on the Main Menu. They are displayed on the screen in the order as listed (left to right, and top to bottom).

Attributes in <MainMenu>

Attribute	Description
ncol	Number of columns in the menu. It is allowed to specify a variable here to track for example, whether the Line Keys are hidden, as: `"$eval($sys.lines?4:6)"` The last example uses 4 columns when the Line Keys are visible and 6 columns when they are not. Default = "3"
width	Pixel width of the table. It is allowed to specify a variable here to track for example, whether the Line Keys are hidden, as: `"$eval($sys.lines?320:480)"` The last example is 320 pixel wide when the Line Keys are visible and 480 pxiel wide when they are not. Default = "320"
height	Pixel height of the table. Default = "204"
xpos	X position of the upper left corner Default = "0"
ypos	Y position of the upper right corner Default = "0"
noscroll	Whether to show a scroll bar if there are more than 1 page of menu items. Value can be 0 (show scrollbar) or 1 (hide scrollbar) Default = "1"
scrollstyle	Scrollbar Style, of scrollbar is enabled. Valid values are: • 0 – slider on the right of the table • 1 – up/down arrow icon on the top and/or bottom of the screen • 2 – A number of dots on the right of the table corresponding to number of pages of menu items; the highlighted dot indiates the current page Default = "1"
c1 – c4	Four constants to substitute respectively @c1 - @c4 in item and style attributes. For example the value can be set to the common root path of the image files to be used as icons for all menu items,

	such as: `c1="http://abcd.com/phone/icons/"`. Then later in an <item> you may specify that icon='@(c1)my-picture.png' Default = ""
cell-width	Pixel width of each menu item cell Default = "80"
cell-height	Pixel height of each menu item cell Default = "68"
hlborder	Whether to draw a border around the highlighted menu item; 0 means no border, > 0 means border thickness Default = "0"
hlbordercolor	Color of the border surrounding the highlighted menu item. Default = "0xffff00"
textcolor	Default text color to use inside all menu items. Default = "0xffffff"
hltextcolor	Default text color to use inside the highlighted menu item. Default = "0xffff00"
font	Default font type to use inside all menu items Default = "@gfont"
size	Default font size to use inside all menu items Default = "10"

Attributes in <MainMenu> <item> Element

Attribute	Description
id	Required for every <item>. It specifies the type of the item. The valid values are:

id	Description
phone-book	The built-in Phone Book app
call-history	The built-in Call History app
settings	The built-in Settings (Configuration) app
preferences	The built-in User Preferences app
calls	The built-in (Current) Calls app
prod-info	The built-in Product Information app
netcf	The built-int Network Services (Configuration) app
netdir	The built-in Network Directory app
buddy	The built-int Buddy List app
sd	An instance of the Speed Dial function similar to a feature key with function = Speed Dial; the *name*, *number*, and *service* attributes in the <item> have the same meaning of those of the feature key
blf	An instance of the BLF function similar to a feature key with function = Busy Lamp Field; the *name*, *number*, and *service* attributes in the <item> have the same meaning of those of the feature key
pres	An instance of the Presence Monitor function similar to a feature key with function = Presence Monitor; the *name*, *number*, and *service* attributes in the <item> have the same meaning of those of the feature key
mwi	An instance of the Message Status function similar to a feature key with function = Message Status; the *name*, *number*, and *service* attributes in the <item> have the same meaning of those of the feature key
cpm	An instance of the Call Park Monitor function similar to a feature key with function = Call Park Monitor; the *name*, *number*, and *service* attributes in the <item> have

		the same meaning of those of the feature key
	acturl	Each instance of the <item> invokes an Action URL (Obihai IP Phone XML APP). It is similar to a feature key with function = Action URL; the *name*, *number*, and *service* attributes in the <item> have the same meaning of those of the feature key

bgimg

Optional. Internal path or URL of a background picture

id	bgimg (default value)		
	Tomas	Ulrik	Colorful
phone-book			
calls			
call-history			
buddy			
prod-info			
settings			
preferences			

	netcf			
	netdir			
	sd blf pres cpm acturl mwi			

hlimg	Optional. Internal path or URL of a background picture when item is highlighted
label	Optional. This is usually the title name of the item on the screen shown at the bottom (below the main icon).

id	label (default)
phone-book	Contacts
calls	Calls
call-history	Call History
buddy	Buddy List
prod-info	Product Info
settings	Settings
preferences	Preferences
netcf	Net Services
netdir	Net Dir
sd	$item.name if not blank, or $item.number if not blank, or Speed Dial
blf	$item.name if not blank, or $item.number if not blank, or BLF
pres	$item.buddy.name if $item.number is not blank, or Presence
cpm	$item.name if not blank, or $item. number if not blank, or Call Park
acturl	$item.name if not blank, or XML APP
mwi	$item.name if not blank, or $iterm.number if not blank, or Message

label2

Optional. This is a smaller label that can be used to show additional information about the menu item. For example, it can show a number that reflects the number of new messages for 'mwi' items

The default is (empty) except for the following item ids:

id	label2 (default)	Description
calls		This text $\{n\}$ with the oval background is shown only when there are one or more active calls on the phone where $\{n\}$ represents the number of active calls. Otherwise it is hidden. Combining this with the default background image for calls, it will look like theses when there are 2 active calls and when there are none, respectively:

			
	call-history	{n}	This text {n} with the oval background is shown only when there are one or more new missed calls where {n} represents the number of new missed calls. Otherwise it is hidden. Combining this with the default background image for call-history, it will look like these when there are 2 new missed calls and when there are none, respectively: 

icon

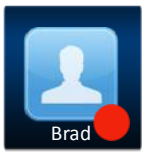
Optional. The default is (empty) except for the following item ids:

id	icon (default)	Description
pres		This icon shown when there is no buddy picture available.
		If the buddy picture is available, it is shown as the icon
cpm		This icon is shown when the state of the parking space (or orbit) is unknown
		This icon is shown when the parking space is available
		The icon is shown when the parking space is occupied
blf		Shown when blf state is offline
		Shown when blf state is idle
		Shown when blf state is ringing
		Shown when blf state is busy (i.e. on the phone)
		Shown when blf state is holding
		Shown when blf state is parked (i.e., a call is parked against the monitored extension)
acturl		
mwi		
sd	Tomas: Ulrik: Colorful: 	

icon2

Optional. The default is (empty) except for the following item ids:

id	icon2 (default)	Description

	mwi		This icon shown when there are new messages in the underlying mailbox. Otherwise the icon is hidden. When combining this with the default mwi background image, it will look like these on the screen when there are new messages and when there are no new messages, respectively: 
	pres		Shown when presence is offline
			Shown when presence is online (i.e. available)
			Shown when presence is dnd (i.e. busy)
			Shown when presence is away
			An example combining default background image, icon, and presence:  In this case the buddy picture is not available and presence is dnd.
style	Specify the *id* of the <style> in the <MainMenuItemStyles> XML to use to style this item.		
hidden	Specify a value that evaluates to "1" to make the item invisible, or "0" to make		
refresh-id	It is possible to push an <ObihaiIPPhoneExecute> XML APP to the phone to trigger the phone to refresh some screen elements. For this purpose, the XML would include one or more <ExecuteItem> with the attribute URI ='RefreshWidget:{*comma-separated-list of refresh-id's*} Here the refresh-id attribute is used to refrence this item for the same purpose		

Macros available in <MainMenu>

Macro	Where	Description
$sys.lines	Anywhere	Returns "1" if Line Keys are shown or "0" if Line Keys are shown
@c*n*, *n* = 1,2,3,4	<item>	The value of the c*n* attribute in <MainMenu>
@gfont	<MainMenu>	
$eval	Anywhere	

Example

```
<MainMenu ncol='$eval($sys.lines?4:6)' width='$eval($sys.lines?320:480)' noscroll='0' scrollstyle='2'
          c1='http://192.168.15.225:8080/phone-icons/main-menu/'
          c2='http://192.168.15.225:8080/icons/'
          c3='http://192.168.15.225:8080/phone-xml/'
          c4='http://192.168.15.225:8080/pictures/'
          cell-width='80' cell-height='68' hlborder='2' hlbordercolor='0xffff00'
          textcolor='0xffffff' hltextcolor='0xffff00'
          font='@gfont' size='10'>
  <item id='phone-book' bgimg='@(c1)phbk.png' label='Contacts'/>
  <item id='calls' bgimg='@(c2)calls.png' label='Calls'/>
  <item id='call-history' bgimg='call-history.png' label='Recent'/>
  <item id='prod-info' bgimg='prod-info.png' label='Product Info'/>
  <item id='settings' bgimg='settings.png' label='Settings'/>
  <item id='preferences' bgimg='preference.png' label='Preferences'/>
  <item id='buddy' bgimg='buddy.png' label='Buddy List'/>
  <item id='netcf' bgimg='net-services.png' label='Net Services'/>
  <item id='netdir' bgimg='net-dir.png' label='Net Dir'/>
  <item id='sd' bgimg='spd.png' number='14089991001' service='sp1' name='Angelina Jolie'
        label='Angelina'/>
```

```
  <item id='sd' bgimg='spd.png' number='14089991002' service='sp2' name='Brad Pitt'
        label='Brad'/>
  <item id='mwi' style='mwi' service='sp1' number='3001' label='VM-3001'/>
  <item id='pres' style='pres' service='sp1' number='jjsm..' name='John Smith' label='JS'/>
  <item id='pres' style='pres' service='sp1' number='rs..' name='Ron Stone' label='Ron'/>
  <item id='pres' style='pres' service='sp1' number='tt123..' name='Tom Tyson' label='Tom T.'/>
  <item id='blf' style='blf' service='sp1' number='3008' label='John'/>
  <item id='blf' style='blf' service='sp1' number='3007' label='Susan'/>
  <item id='blf' style='blf' service='sp1' number='3008' label='Samuel'/>
  <item id='acturl' style='acturl' number='@(c3)weather.xml' label='Weather' refresh-id='weather'/>
  <item id='cpm' service='sp1' number='71' label='71'/>
</MainMenu>
```

<MainMenuItemStyles> XML

General structure:

```
<MainMenuItemStyles>
  <style id="abc" ...>
    <label .../>
    <label2 .../>
    <icon .../>
    <icon2 .../>
  </style>
  ...
  <style id='yz' ...>
    ...
  </style>
</MainMenuItemStyles>
```

Elements in <MainMenuItemStyles>

Element	Attributes	Description
<MainMenuItemStyles>	Required: Optional:	Root element. This element has no attributes defined
<style>	Required: id Optional: bgimg, hlimg, bgcolor, hlcolor, hidden, scaling,	The attributes in this element specifies the style to render the background of the referencing main menu item's screen tile.
<label>	Required: Optional: bgimg, bgcolor, hidden, hidden-if-empty, xpos, ypos, width, height, align, valign, font, size, textcolor, hltextcolor, wrap, resize	The value is the primary text to display on the tile
<label2>	Required: Optional: bgimg, bgcolor, hidden, hidden-if-empty, xpos, ypos, width, height, align, valign, font, size, textcolor, hltextcolor, wrap, resize	The value is the secondary text to display on the tile
<icon>	Required: Optional: hidden, xpos, ypos, width, height, align, valign, resize, aspect	The value is internal file path or http/https URL of the primary icon to display on the tile
<icon2>	Required:	The value is internal file path or http/https URL of the

	Optional: hidden, xpos, ypos, width, height, align, valign, resize, aspect	secondary icon to display on the tile

Attributes in <MainMenuItemStyles>

Attribute	Where	Description
id	<style>	The id to reference this style in the *style* attribute of a <MainMenu> <item>. If *style* is not specified in <item>, then by default it references the <style> with id = "default".
bgimg	<style>, <label>, <label2>, <icon>, <icon2>	When appears in <style>, it is the background image for the referencing main menu item's screen tile. When appears in <label>, <label2>, <icon>, or <icon2> (though less common), it is the background image to cover the bounding box of the respective widget
hlimg	<style>	The background image for the referencing main menu item's screen tile, when the item is highlighted. If hlimg is not specified, the same bgimg is shown whether the item is highlighted or not.
bgcolor	<style>, <label>, <label2>, <icon>, <icon2>	When appears in <style>, it is the background color or color pattern of the referencing main menu item's screen tile. If bgimg is defined and non-empty and covers the entire item, then bgcolor would be invisible. If bgimg has some transparent area or does not cover the entire item, then bgcolor will show through those uncovered area, if specified. Examples: "0xffffff,0xaaaaaa,1" (gradient between two given color) or "0x4546ff" (solid with the same color everywhere) When appears in <label>, <label2>, <icon>, or <icon2> (though less common), it is the background color or color pattern to cover the bounding box of the respective widget.
hlcolor	<style>	When appears in <style>, it is the background color or color pattern of the referencing main menu item's screen tile on screen, when the item is highlighted. If hlcolor is not specified, the same bgcolor is shown whether the item is highlighted or not.
xpos, ypos, width, height	<icon>, <icon2>, <label>, <label2>	These 4 attributes specify the upper left corner pixel position (relative to the parent bounding box), and the pixel width and height of the bounding box enclosing the element. Default for xpos is "0", ypos is "0", width and height are same as the parent bounding box
align	<icon>, <icon2>, <label>, <label2>	Specifies the horizontal alignment of the picture within the icon's bounding box (note: not the parent's). Value should be "left", "right", or "center"
valign	<icon>, <icon2>, <label>, <label2>	Specifies the vertical alignment of the picture within the icon's bounding box (note: not the parent's). Value should be "top", "bottom", or "center"
src	<icon>, <icon2>	Overwrites the file path or URL of the icon in the referencing main menu item. It can be an internal storage path or a http/https URL
font	<label>, <label2>	The font to use to render text for the widget (value must be either "@gont" or "@gfont-bold")
size	<label>, <label2>	The font size to use to render text for the widget
wrap	<label>, <label2>	Specifies whether to wrap ('1") or truncate ("0") the text when it cannot fit the label's bounding box
resize	<label>, <label2>, <icon>,	For labels, specifies whether to resize ("1") or not ("0") the

	<icon2>	text when it cannot fit the labels' bounding box. For icons, spcecifies whether to resize ("1") or not ("0") the picture when it cannot fit the fit the icon's bounding box.
aspect	<icon>, <icon2>	Spcecifies whether to maintain aspect ratio ("1") or not ("0") when resizing the picture to fill the icon's bounding box
scaling	<style>	Spceifies how to scale the background image if one is defined for the style
textcolor	<label>, <label2>	The textcolor to use to render the text for the label. For example "0xffffff" (white)
hltextcolor	<label>, <label2>	The textcolor to use to render the text for the label when the referencing item is highlighted. For examole "0xffff00" (yellow)
hidden	<style>, <label>, <label2>, <icon>, <icon2>	When appears in <style>, specifies whether the referencing item should be hidden ("1") or not ("0") When appears in <label>, <label2>, <icon>, or <icon2>, specifies whether the respective should be hidden ("1") or not ("0")
hidden-if-empty	<lable>, <label2>	Specifies if the label should be hidden ("1") or not ("0") if the text string it contains is empty

Macros available in <MainMenuItemStyles>

Attribute	Where	Description
@gfont	<style> and all its childs	The default font family configured on the phone
$item.name	<style> and all its childs	The name attribute of the <item> referencing the <style>
$item.number	<style> and all its childs	The number attribute of the <item> referencing the <style>
$item.mwi.value	<style> and all its childs	The MWI status value *n* of the <item> referencing the <style> if the item has the attribute id="mwi", where *n* is an integer greater than 0 if there are new messages waiting, or 0 if there are none
$item.blf.value	<style> and all its childs	The BLF status value of the <item> referencing the <style> if the item has the attribute id="blf", where the value can be one of the following: - `idle` for no calls - `trying` for attempting outgoing call - `peerring` for outgoing call ringing - `connected` for incoming or outgoing call connected - `holding` for holding a call - `ended` for call ended - `parked` for a call parked against the extension - `ring` for incoming call ringing
$item.buddy.presence	<style> and all its childs	The buddy's presence value if the <item> referencing the <style> has the attribute id="pres", where the value can be one of the following: - `offline` - `online` - `dnd` - `away` - `xa`
$item.buddy.picture	<style> and all its childs	The internal path that stores the buddy's picture if the <item> referencing the <style> has the attribute id="pres".
$item.buddy.name	<style> and all its childs	The buddy's name if the <item> referencing the <style> has

		the attribute id="pres".
$item.buddy.number	<style> and all its childs	The buddy's telephone number if the <item> referencing the <style> has the attribute id="pres".
$item.buddy.status	<style> and all its childs	The buddy's status (an arbitrary string) if the <item> referencing the <style> has the attribute id="pres".
$eval	<style> and all its childs	

Example

```
<MainMenuItemStyles>
  <style id='default' hlimg=''>
    <label align='center' xpos='0' width='80'/>
    <label2 hidden='1'/>
    <icon hidden='1'/>
    <icon2 hidden='1'/>
  </style>
  <style id='pres' bgimg='@imgdir/pres_mm.png' hlimg=''>
    <label align='center' font='@gfont-bold' textcolor='0xffffff' hltextcolor='0xffff00'/>
    <label2 hiddend='1'/>
    <icon align='center' width='44' height='44' align='center'
          src='$eval($item.buddy.picture!=?
              /media/ram/$item.buddy.picture:@imgdir/mm_person.png)'/>
    <icon2 xpos='50' ypos='38' width='16' height='16'
           src='@imgdir/mm_pres_$(item.buddy.presence).png'/>
  </style>
</MainMenuItemStyles>
```

Title Bar Customization

The Title Bar can be customized by specifying a valid <TitleBarStyle> XML in the **TitleBarStyle** parameter. The general structure is:

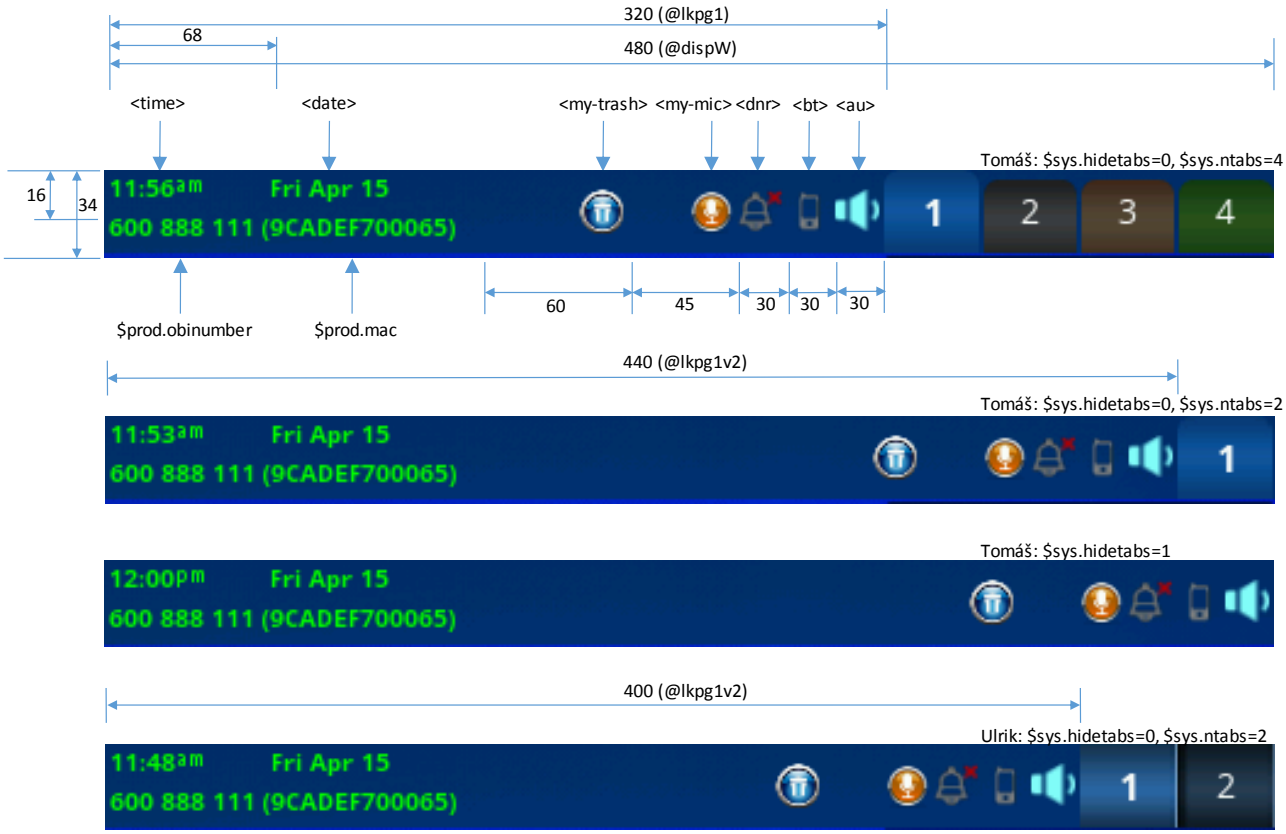
```
<TitleBarStyle attributes... >
  <!-- 0 or more widgets: such as time, date, text, img, ... -->
  <Notifications attributes... >  <!-- optional -->
    <!-- 0 or more notification elements, such as dnd, cfa, ... -->
  </Notifications>
  <LineKeyTabs attributes... >    <!-- optional -->
    <tab-style attributes... >
  </LineKeyTabs>
</TitleBarStyle>
```

Here is an example:

```
<TitleBarStyle size='10' textcolor='green' font='@gfont-bold'>
  <time xpos='2' format="@timeFmt" />
  <date xpos='68' format="@dateFmt" />
  <text xpos='2' ypos='16'>$prod.obinumber ($prod.mac)</text>
  <Notifications list='*,my-mic,my-trash'
                 width='$eval($sys.hidetabs?@dispW,($sys.ntabs==4)?@lkpg1,@lkpg1v2)'
                 align='right' xpos='0'
                 base-url='http://192.168.15.149/icons/'>
    <my-trash icon='trash.png' refresh-id='mytrash' xspace='60'/>
    <my-mic icon='mic.png' refresh-id='mymic' xspace='45'/>
  </Notifications>
</TitleBarStyle>
```

This simple example produces a title bar similar to ones shown below: The first picture shows the result when there are 4 Line Key Tabs with the Tomáš skin. The third picture shows the result when the Line Key Tabs are hidden, with the Tomáš Skin. The fourth picture shows the result when there are 2 Line Key Tabs, with the Ulrik Skin. Skin is configured by setting the ***User Preferences***::**Skin** parameter, number of Line Key Tabs (2 or 4) is configured by setting the ***User***

***Preferences*::LineKeyMaxTabs** parameter, and hiding the Line Key Tabs altogether is controlled by the setting of the ***User Preferences*::LineKeyTabs** option. The notification area is divided into a number of non-overlapping packed *holding boxes*, with each box holding one displayed icon and a box width equals to the value of the *xspace* attribute of the corresponding notification element.



The following table provides a description of the elements used in the last example:

Element	Description
<TitleBarStyle>	The root element. The attributes used are inherited by all the descendants it contains: - size = the font-size for rendering text - textcolor = the color for rendering text - font = the font to render text with
<time>	The time widget which shows the current local time. The time format follows the setting of the ***User Preferences*::TimeFormat** parameter. Attributes used: - xpos = x pixel-position to start rendering the current time (as text) By default: xpos and ypos attributes are 0 if not specified
<date>	The time widget which shows the current local time. The time format follows the setting of the ***User Preferences*::DateFormat** parameter. Attributes: - xpos = x pixel-position to start rendering the current date (as text) By default: xpos and ypos attributes are 0 if not specified
<text>	Arbitrary text. Attributes used: - xpos = x pixel-position to start rendering the text - ypos = y pixel-position to start rendering the text Variables/Constants used: - $prod.obinumber = the 9-digit OBi number of the phone - $prod.mac = the MAC address of the phone

	By default: xpos and ypos attributes are 0 if not specified
<Notifications>	Specifies and contains a list of notification child elements whose icons are to be shown on the title bar in the given order. Attributes used: - list = a comma separated list of notifications to show on the title bar. A * includes all the built-in notifications - width = the total pixel width of the notication area on the title bar - align = specifies whether the icons should align to the right side of the notification area or to the left side of the notification area. "right" is the default if this attribute is not specified. With right alignment, the icons are drawn in right to left order. With left alignment, the icons are drawn in left to right order - base-url = A string to prepend to the icon attribute in each child element to be used as URL to get the icon image Variables/Constants used: - $sys.hidetabs = 0 if the LineKeyTabs option is enabled, or 1 otherwise - $sys.ntabs = the value of LineKeyMaxTabs (2 or 4) - @dispW= the pixel width of the LCD: 480 for OBi1032/1062, 320 for OBi1022, and 800 for OBi2000 Series - @lkpg1 = the x-position of Line Key Tab 1 - @lkpg1v2 = the skin-dependant x-position of Line Key Tab 1
<my-trash>	An example of a user-defined notification child element. Attributes used: - icon = the url to download the icon. The actual URL is formed by prepending this attribute with the value of the base-url. For this example, the full url is http://192.168.15.149/icons/transh.png - refresh-id = Anarbitray but unique string among all referesh-id's to be used as reference to trigger a redraw this icon by an external application - xpace = x-pixel width of the holding box that holds the icon, which is drawn aligned with the left side of the holding box if <Notifications> has align="left", or with the right side of the holding box otherwise. By default: xspace = 30
<my-mic>	Another example of a user-defined notification child element, similar to <my-trash>

To see the effect of a left-aligned <Notifications>, this above example can be modified slighty by changing the value of the *align* attribute from "right" to "left" and the xpos attribute from "0" to "150":

```
<TitleBarStyle size='10' textcolor='green' font='@gfont-bold'>
  <time xpos='2'/>
  <date xpos='68'/>
  <text xpos='2' ypos='16'>$prod.obinumber ($prod.mac)</text>
  <Notifications list='*,my-mic,my-trash'
                 width='$eval($sys.hidetabs?@dispW,($sys.ntabs==4)?@lkpg1,@lkpg1v2)'
                 align='left' xpos='150'
                 base-url='http://192.168.15.149/icons/'>
    <my-trash icon='trash.png' refresh-id='mytrash' xspace='60'/>
    <my-mic icon='mic.png' refresh-id='mymic' xspace='45'/>
  </Notifications>
</TitleBarStyle>
```

The picture below shows the corresponding result (with 4 Line Key Tabs and the Tomáš skin):

<TitleBarStyle> Attributes

All attributes are optional. When specified, the attributes are inherited by the child widgets, such as size (for text size), textcolor, font, etc. The only exception is the bgcolor attribute, which is applied to the Title Bar itsef to overwrite the default background color. Here is an example of using bgcolor attribute:

```
  <TitleBarStyle size='10' bgcolor='0x444444,white,0'
                           textcolor='blue' font='@gfont-bold'>
    <time xpos='2'/>
    <date xpos='68'/>
    <text xpos='2' ypos='16'>600 888 111 ($prod.mac)</text>
```

```
  <Notifications list='*,my-mic,my-trash'
                 width='$eval($sys.hidetabs?@dispW,($sys.ntabs==4)?@lkpg1,@lkpg1v2)'
                 align='right' xpos='0'
                 base-url='http://192.168.15.149/icons/'>
    <my-trash icon='trash.png' refresh-id='mytrash' xspace='60' width='34' height='34'/>
    <my-mic icon='mic.png' refresh-id='mymic' xspace='45' width='35' height='35' />
  </Notifications>
</TitleBarStyle>
```

Below is the result of rendering the above XML (with 4 Line Key Tabs and the the Tomáš skin):

The <Notifications> Element

The general structure of the element:

```
<Notifications list='a,b,c,d,...' width='200' align='right' xpos='0'
               base-url='http://myserver.abcd.com/icons/'>
  <a icon='a-icon.png' refresh-id='a-status' xspace='40'/>
  <b icon='b-icon.png' refresh-id='b-status' xspace='35'/>
  <c icon='c-icon.png' refresh-id='c-status' xspace='30'/>
  <d icon='d-icon.png' refresh-id='d-status' xspace='30'/>
  <!-- ... -->
</Notifications>
```

A description of the supported attributes in the element:

<Notifications> Attribute	Description
list	A comma separated list of the names of the notification elements whose icons are to be drawn on the Title Bar in the notifications area, and in the order as listed. A * element name includes all the built-in elements (in the built-in order). For a built-in notification element name in the list (* or explicitly named), it is optional to include a child element of the same name; the icon will be displayed according to the built-in attributes if the corresponding child element is not given. For user-defined notification element name in the list, there should be a corresponding child element of the same name that describes where to get the user-defined icon image and how it should be displayed. Note that all valid icons are packed in the notification area; no space reserved for invalid icons or undefined notification elements.
width	Defines the pixel width of entire notification area on the Title Bar
align	Either "left" or "right". Specifies if the displayed icons should be left aligned or right aligned in the notification area. Icons are drawn from left to right if left aligned, or right to left if right aligned
xpos	Defines the starting x pixel position of the notification area, with respect to the left edge of the LCD
base-url	The base-url to be inherited by all child element. The value is prepended to the icon attribute in each child element to form the complete URL (or file path) to get the icon image

The <Notifications> element contains a number of child elements, each with a unique element name that matches one of the members in the *list* attribute and specifies an icon to show in the notification area of the title bar. There are a number of built-in notification elements with their names and meaning listed in the following table:

Built-in <Notifications> Element	Description	Values of $state
au	Audio device status: the icon shows which audio device is active, if any	none: No active audio (phone is idle) bt: Bluetooth headset audio is active handset: Handset audio is active speaker: Speakephone audio is active headset: Headset audio is active rj9: RJ9 headset audio is active trs: 3.5mm Headset audio is active

		Note: $state is alias of $sys.auPath

bt	Bluetooth connection status if Bluetooth is enabled on the phone	Connected To (note: includes 1 white space at the end) = Connected to a Bluetooh device
		Trying To Connect With (note: includes 1 white space at the end) = Trying to connect to a Bluetooth device
		Disconnected = Not connected to any device
		No Home/Roam Network Available = Bluetooth offline
		
		Note:
		- $state is alias of $bt.status
		- $bt.hsp is the pairing mode:
		  o Enabled = Pair with headset
		  o Disabled = Pair with mobile phone

cfa	Call forward unconditional service on/off status	Enabled = feature is enabled Disabled = feature is disabled
dnd	Do Not Disturb setting on/off status	Enabled = feature is enabled Disabled = feature is disabled
dnr	Do Not Ring setting on/off status	Enabled = feature is enabled Disabled = feature is disabled
mwi	Mailbox Status	0 = no new messages in mailbox *n* (where *n* > 0) = New messages in mailbox
net	Ethernet Status: Shows an icon to indicate if Ethernet is not connected properly	0 = Ethernet down (no link or no IP address) *n* (where *n* > 0) = Ethernet is up
rpd	Repeat Dialing status: Shows an icon to indicate if (auto) repeat dialing is in progress	0: Repeat dialing is not not active 1: Repeat dialing is active
dialxfer	Dialing a number to transfer: Shows an icon to indicate dialing a transfer target number	0 = not dialing a blind transfer target 1 = dialing a blind transfer target Note: - $state is alias of $dial.bxfer - $dial.divert = 1 when dialing a diversion target number - $dial.xfer = 1 when dialing a transfer target number - $dial.conf = 1 when dialing a conference target number - At most one of $dial.xfer, $dial.bxfer, dial.divert, and dial.conf can be 1
dialtype	Current dialing type: Shows an icon to indicate if the user is dialing a special number. For example: dialing an intercom, dialing a call-forward number, making an anonymous call, requesting to barge-in the called party, requesting to whisper (a.k.a. coach) the called party.	cfwd = Entering a cal forward number spd = Entering a Speed Dial 99 entry intercom = Entering a target number to intercom barge-in = Entering a target number to barge-in coach = Entering a target number to coach (a.k.a. whisper) pickup = Enter a target number to pickup (directed call pickup) bci = Dialing an anonymous call (block caller ID) Note: $state is alias of $dial.type
rbt	Reboot Needed status: Shows an icon to indicate if a reboot is needed to have a recent configuration change to take effect	0: Reboot not needed 1: Reboot needed for some recent changes to take effect
upg	Firmware Update status: Shows an icon that indicates if firmware update is in progress	0: software upgrade is not in progress 1: software upgrade in progress
wifi	WiFi connection status: Shows the signal strength or connection status if WiFi is enabled	fail = Not connected to any AP busy = Trying to connect to an AP none or *empty* = WIFI disable sec = Connected to a secured AP open = Connected to an open-access (insecure) AP Note: - $state is alias of $wifi.status - $wifi.signal is the WIFI signal strength (0-5)
zt	ZT Customization status: Shows an icon to indicate the ZT Customization process has not yet completed on the phone	0 = ZT customization has not been completed *n* (where *n* > 0) = ZT customization has been completed
usbkb	USB Keyboard status: Shows a keyboard icon to indicate that a USB keyboard is connected to the phone	0 = no USB keyboard connected 1 = USB keyboard connected Note: - $state is alias ofr $sys.usbkb - $sys.usbkbcaps = 1 if keyboard caps-lock is on, or 0 otherwise

It is optional to include a notification element for a built-in notification element (such as <dnd> or <net>) inside <Notifications>. But if you do explicitly include one, you can overwrite the display properties of the built-in notification element (for instance, to change the icon to display for the <dnd> element when the dnd feature is enabled or disabled). You can add additional user-defined notification elements with unique names that do not conflict with the built-in notification elements.

Each notification element may contain the following optional attributes:

<Notifications > Child Element Attributes	**Description**
icon	This value is prepended with the value of base-url to form the URL or file path of the icon image to display. It can be an external (http:// or https:// URL) or an internal path (filename). If not specified, the default icon will be shown for the built-in notification element, but no icon will be shown for user-defined notification element.
xspace	The width of the holding box that holds the icon of this element. For left aligned <Notifications>, the icon is drawn with its bounding box aligned with the left edge of the holding box. Otherwise, the icons is drawn with its bounding box aligned with the right edge of the holding box
base-url	Overwrites the base-url value inherited from the parent
ypos	Y pixel-position of the upper left corner of the icon to display
width	Pixel width of the bounding box of the icon
height	Pixel height of the bounding box of the icon
resize	Whether to resize ("1") or truncate ("0") the icon to fit the given bounding box
aspect	Whether to maintain aspect ratio ("1") or not ("0") when resizing the icon to fit the given bounding box
align	Horizontal alignment w.r.t. its bouding box to draw the icon
valign	Vertical alignment w.r.t. its bounding box to draw the icon
border	Whether to draw a border around the bounding box of the icon
bordercolor	The color of the border to draw around the bounding box of the icon
refresh-id	Reference ID to use to trigger a redraw of the icon by an external Obihai IP Phone XML application

As hinted above, you can use the $state variable inside an $eval macro to conditional set the icon of a built-in notification element. For example, you can overwrite the icon for the <dnd> element by specifying the icon attribute in that element as:

```
<dnd icon="$eval(($state==Enabled)?{my-dnd-on.png}:{my-dnd-off.png})" xspace="36" width="32" height="32"/>
```

The <LineKeyTabs> Element

This element is for the customization of the Line Key Tabs area of the Title Bar (which holds all of the 1, 2 or 4 tabs depending on the user setting, skin, and phone model). In the following example, we replace the default background picture of the tab area (which is a single picture) with a simple gradient color, and highlight the tab number of the current tab.

```
<LineKeyTabs bgimg='' bgcolor='white,red'>
  <tab-style valign='top' border='1' bordercolor='black'
             font='$eval(($lkeys.page==@p)?{@gfont-bold}:{@gfont})'
             textcolor='white'
             size='$eval(($lkeys.page==@p)?16:12)'/>
</LineKeyTabs>
```

The result of rendering the above example is shown below (with 4 Line Key Tabs and the the Tomáš skin):

Here is a description of the elements/attributes used in the last example:

Element	**Description**
<LineKeyTabs>	Attributes used: - bgimg: An empty value is used to clear the background image (so that background color can be shown) - bgcolor: Specifies the background color as a gradient color that varies from white to red vertically (default) -
<tab-style>	Attributes used: - valign: Align the text in each tab (the tab number) to the top edge of the bounding box

	- border: Draw a 1-pixel width border around the bounding box - bordercolor: Draw the border in black - font: Specifies to use bold face font (to render the tab number) when the tab number (@p) matches the current tab ($lkeys.page), otherwise use simple face font - textcolor: Render text (tab number) in white - size: Use textsize 16 when the tab number (@p) matches the current tab ($lkeys.page), otherwise use textsize 12

Here is another example on how to reduce the width the Line Key Tabs area, thus making more room available to display the notification icons:

```
<TitleBarStyle size='10' bgcolor='0x444444,white,0' textcolor='blue' font='@gfont-bold'>
  <time xpos='2'/>
  <date xpos='68'/>
  <text xpos='2' ypos='16'>600 888 111 ($prod.mac)</text>
  <Notifications
        list='*,my-mic,my-trash'
        width='$eval($sys.hidetabs?@dispW,($sys.ntabs==4)?(@dispW-96),(@dispW-24))'
        align='right' xpos='0' base-url='http://192.168.15.149/icons/'>
    <my-trash icon='trash.png' refresh-id='mytrash' xspace='60' width='34' height='34'/>
    <my-mic icon='mic.png' refresh-id='mymic' xspace='45' width='35' height='35' />
  </Notifications>
  <LineKeyTabs bgimg='' bgcolor='white,red' xpos='$eval(@dispW-96)' width='96'>
    <tab-style xpos='$eval((@p-1)*24)' width='24'
               valign='top' border='1'
               bordercolor='black'
               font='$eval(($lkeys.page==@p)?{@gfont-bold}:{@gfont})'
               textcolor='white'
               size='$eval(($lkeys.page==@p)?16:12)'/>
  </LineKeyTabs>
</TitleBarStyle>
```

And below is the result of rendering the above XML. Note that the xpos in <tab-style> is relative to the left side of Line Key Tabs area, while the xpos in <LineKeyTabs> is relative to the left side of the Title Bar (same as the left side of the LCD).

Trigger update of a notification icon with an ObihaiIPPhoneExecute XML

You can push an <ObihaiIPPhoneExecute> XML to the phone (using SIP/NOTIFY or HTTP/Port) to trigger a redraw of a notification icon. Here is a simple example:

```
<ObihaiIPPhoneExecute beep='yes'>
  <ExecuteItem base-url='http://192.168.15.149/icons/'
               URL='ClearDownloadedDataCache:trash.png'/>
  <ExecuteItem URL='RefreshWidget:mytrash'/>
</ObihaiIPPhoneExecute>
```

In this XML, the first <ExecuteItem> tells the phone to clear the cache entry for the icon that it gets from the URL **http://192.168.15.149/icons/trahs.png**. This is just to make sure the phone will fetch the icon again from the server when redrawing the icon. The second <ExecuteItem> tells the phone to redraw the icon with the reference refresh-id **mytrash**.

Refer this article for more information on writing Obihai IP Phone XML and how to push the XML to the phone.

LED Pattern Customization

Feature Key LED patterns for each well-defined state can be customized using the LED Settings in the configuration. Each customizable pattern is configured in its own parameter as a comma separated list of `{Color}[{Duration}]` pairs, where `{Color}` = `R` (for red), `G` (for green), `O` (for orange), or `X` (for off). The optional `{Duration}` part is the number of milliseconds to show the given color. If `{Duration}` is not specified, the LED stays at the given color indefinitely. The entire pattern is played repeatedly from left to right indefinitely until the state changes. Here are some examples:

`X` – Steady off

`R` – Steady red

`R500,X500` – 500 ms red followed by 500 ms off

`G50,X50,G50,X1000` – 50 ms green, 50 ms off, 50 ms green, 1s off

LED Settings Parameters

Parameter Group	Parameter	Description
Call State For a bounded or unbounded Call Key. The LED pattern reflects the local call state.		
IP Phone – LED Settings – Call State	**Dialing**	
IP Phone – LED Settings – Call State	**Trying**	
IP Phone – LED Settings – Call State	**PeerRinging**	
IP Phone – LED Settings – Call State	**Connected**	
IP Phone – LED Settings – Call State	**Ringing**	
IP Phone – LED Settings – Call State	**Holding**	
IP Phone – LED Settings – Call State	**Error**	
IP Phone – LED Settings – Call State	**CallParked**	
IP Phone – LED Settings – Call State	**ServiceDown**	
SCA State For a Call Key that is bounded to a share line. The LED pattern reflects the SCA state when the user of the SCA is on another phone.		
IP Phone – LED Settings – SCA State	**Seized**	
IP Phone – LED Settings – SCA State	**Trying**	
IP Phone – LED Settings – SCA State	**PeerRinging**	
IP Phone – LED Settings – SCA State	**Connected**	
IP Phone – LED Settings – SCA State	**Held**	

IP Phone – LED Settings – SCA State	**PrivateHeld**	
IP Phone – LED Settings – SCA State	**ServiceDown**	
BLF State For feature key that is assigned the function Busy Lamp Field. The LED reflects the state of the monitored entity.		
IP Phone – LED Settings – BLF State	**Idle**	
IP Phone – LED Settings – BLF State	**CallParked**	
IP Phone – LED Settings – BLF State	**Ringing**	
IP Phone – LED Settings – BLF State	**Busy**	
IP Phone – LED Settings – BLF State	**Holding**	
IP Phone – LED Settings – BLF State	**ServiceDown**	
Service State For feature key that is assigned the function Line Monitor. The LED pattern reflects the state of the bounding voice service.		
IP Phone – LED Settings – Service State	**Idle**	
IP Phone – LED Settings – Service State	**InUse**	
IP Phone – LED Settings – Service State	**Ringing**	
IP Phone – LED Settings – Service State	**Holding**	
IP Phone – LED Settings – Service State	**ServiceDown**	
ACD Agent State For feature key that is assigned the function ACD Sign On/Off. The LED reflects the current ACD Agent state.		
IP Phone – LED Settings – ACD Agent State	**LoggedOff**	
IP Phone – LED Settings – ACD Agent State	**Available**	
IP Phone – LED Settings – ACD Agent State	**Unavailable**	
IP Phone – LED Settings – ACD Agent State	**WrappingUp**	
IP Phone – LED Settings – ACD Agent State	**ServiceDown**	
Presence State For feature key that is assigned the function Presence Monitor. The LED reflects the presence of the monitored entity.		
IP Phone – LED Settings – Presence State	**Offline**	
IP Phone – LED Settings – ACD Agent State	**Online**	

IP Phone – LED Settings – ACD Agent State	**Busy**	
IP Phone – LED Settings – ACD Agent State	**Away**	
IP Phone – LED Settings – ACD Agent State	**ExtendedAway**	
IP Phone – LED Settings – ACD Agent State	**ServiceDown**	
Feature Key State Miscellaneous feature key functions not covered in the above.		
IP Phone – LED Settings – Feature Key State	**AnonymousCallEnabled**	For feature key assigned the function Block Caller ID. This is the LED pattern when the feature is enabled.
IP Phone – LED Settings – Feature Key State	**AnonymousCallDisabled**	For feature key assigned the function Block Caller ID. This is the LED pattern when the feature is disabled.
IP Phone – LED Settings – Feature Key State	**AnonymousCallServiceDown**	For feature key assigned the function Block Caller ID. This is the LED pattern when the service that provides the feature is down, if the feature is provided by the ITSP.
IP Phone – LED Settings – Feature Key State	**AnonymousCallBlockEnabled**	For feature key assigned the function Block Anonymous Call. This is the LED pattern when the feature is enabled.
IP Phone – LED Settings – Feature Key State	**AnonymousCallBlockDisabled**	For feature key assigned the function Block Anonymous Call. This is the LED pattern when the feature is disabled.
IP Phone – LED Settings – Feature Key State	**AutoAnswerIntercomEnabled**	For feature key assigned the function Auto Answer Intercom. This is the LED pattern when the feature is disabled.
IP Phone – LED Settings – Feature Key State	**AutoAnswerIntercomDisabled**	For feature key assigned the function Auto Answer Intercom. This is the LED pattern when the feature is disabled.
IP Phone – LED Settings – Feature Key State	**CallForwardEnabled**	For feature key assigned the function Call Forward. This is the LED pattern when the feature is enabled.
IP Phone – LED Settings – Feature Key State	**CallForwardDisabled**	For feature key assigned the function Call Forward. This is the LED pattern when the feature is disabled.
IP Phone – LED Settings – Feature Key State	**CallForwardServiceDown**	For feature key assigned the function Call Forward. This is the LED pattern when the service providing this feature is down.
IP Phone – LED Settings – Feature Key State	**CallParkYes**	For feature key assigned the function Call Park Monitor. This is the LED pattern when there is a call parked on that park orbit.
IP Phone – LED Settings – Feature Key State	**CallParkNo**	For feature key assigned the function Call Park Monitor. This is the LED pattern when there is no call parked on that park orbit.
IP Phone – LED Settings – Feature Key State	**CallParkMonitorServiceDown**	For feature key assigned the function Call Park Monitor. This is the LED pattern when the underlying call park monitoring service is down
IP Phone – LED Settings – Feature Key State	**CallWaitingEnabled**	For feature key assigned the function Call Waiting. This is the LED pattern when the feature is enabled.
IP Phone – LED Settings – Feature Key State	**CallWaitingDisabled**	For feature key assigned the function Call Waiting. This is the LED pattern when the feature is disabled.
IP Phone – LED Settings – Feature Key State	**DoNotDisturbEnabled**	For feature key assigned the function Do Not Disturb. This is the LED pattern when the feature is enabled.
IP Phone – LED Settings – Feature Key State	**DoNotDisturbDisabled**	For feature key assigned the function Do Not Disturb. This is the LED pattern when the feature is disabled.
IP Phone – LED Settings – Feature Key State	**DoNotRingEnabled**	For feature key assigned the function Do Not Ring. This is the LED pattern when the feature is enabled.
IP Phone – LED Settings – Feature Key State	**DoNotRingDisabled**	For feature key assigned the function Do Not Ring. This is the LED pattern when the feature is disabled.
IP Phone – LED Settings – Feature Key State	**ExecFilterEnabled**	For feature key assigned the function Exec Filter On/Off. This is the LED pattern when the feature is enabled.
IP Phone – LED Settings – Feature Key State	**ExecFilterDisabled**	For feature key assigned the function Exec Filter On/Off. This is the LED pattern when the feature is enabled.

IP Phone – LED Settings – Feature Key State	**ExecFilterServiceDown**	For feature key assigned the function Exec Filter On/Off. This is the LED pattern when the service providing this feature is down.
IP Phone – LED Settings – Feature Key State	**ExecAssistEnabled**	For feature key assigned the function Exec Assistant. This is the LED pattern when the assistant is not filtering calls for any executives (i.e., the executive list for this assistant is empty).
IP Phone – LED Settings – Feature Key State	**ExecAssistDisabled**	For feature key assigned the function Exec Assistant. This is the LED pattern when the assistant is filtering calls for at least one executive.
IP Phone – LED Settings – Feature Key State	**ExecAssistDivertOn**	For feature key assigned the function Exec Assistant. This is the LED pattern when the assistant has turned on the Divert option
IP Phone – LED Settings – Feature Key State	**ExecAssistServiceDown**	For feature key assigned the function Exec Assistant. This is the LED pattern when the service providing the Exec Assistant feature is down.
IP Phone – LED Settings – Feature Key State	**HotelingGuestLoggedOn**	For feature key assigned the function Hoteling. This is the LED pattern when a guest has logged on.
IP Phone – LED Settings – Feature Key State	**HotelingGuestLoggedOff**	For feature key assigned the function Hoteling. This is the LED pattern when no guest has logged on.
IP Phone – LED Settings – Feature Key State	**HotelingServiceDown**	For feature key assigned the function Hoteling. This is the LED pattern when the service providing this feature is down.
IP Phone – LED Settings – Feature Key State	**NewMessagesWaitingYes**	For feature key assigned the function Message Status. This is the LED pattern when there are new messages in that mailbox.
IP Phone – LED Settings – Feature Key State	**NewMessagesWaitingNo**	For feature key assigned the function Message Status. This is the LED pattern when there are no new messages in that mailbox.
IP Phone – LED Settings – Feature Key State	**MWIServiceDown**	For feature key assigned the function Message Status. This is the LED pattern when the MWI service for that mailbox is down.
IP Phone – LED Settings – Feature Key State	**PageGroupJoined**	For feature key assigned the function Page Group 1 or Page Group 2. This is the LED pattern when the phone has joined the group.
IP Phone – LED Settings – Feature Key State	**PageGroupLeft**	For feature key assigned the function Page Group 1 or Page Group 2. This is the LED pattern when the phone has left the group.
IP Phone – LED Settings – Feature Key State	**PageGroupMeTalking**	For feature key assigned the function Page Group 1 or Page Group 2. This is the LED pattern when the user is talking to the group.
IP Phone – LED Settings – Feature Key State	**PageGroupThemTalking**	For feature key assigned the function Page Group 1 or Page Group 2. This is the LED pattern when the user of the phone is listening and someone else in the group is talking.
IP Phone – LED Settings – Feature Key State	**SecurityClassServiceDown**	For feature key assigned the function Security Class. This is the LED pattern when the service providing this feature is down.
VMWI Lamp		
IP Phone – LED Settings – VMWI Lamp	**NewMessagesWaitingYes**	New messages for the phone from all mailboxes
IP Phone – LED Settings – VMWI Lamp	**NewMessagesWaitingNo**	No new messages for the phone from all mailboxes
IP Phone – LED Settings – VMWI Lamp	**Disabled**	VMWI is disabled
IP Phone – LED Settings – VMWI Lamp	**Ringing**	Describe whether the LED should blink with the given pattern when the phone is ringing. Note that if the pattern is blank, ringing will not affect the VMWI Lamp

GUI Menu Customization

The parameters for customizing some GUI menus are summarized in the following table. Each menu parameter is a comma separated list of menu items, where each *menu item* is specified with an *item id* followed by an optional semicolon and an *item display text* separated by a semicolon. That is:

menu = *item, item, …, item*

item = *item-id;item-display-text*

When there are multiple instances of the same menu parameters are specified, the items will concatendated together into a single item list internally. Items are displayed in the order they are specified. If display-text is not specified for an item, the default displayed text will be used.

Parameter Group	Parameter	Description
Preferences Menu		
IP Phone – Phone Settings – GUI Menus	**PreferencesMenu1**	For example: `language;Language,timeFormat;Time Format,dateFormat;Date Format`
IP Phone – Phone Settings – GUI Menus	**PreferencesMenu2**	
IP Phone – Phone Settings – GUI Menus	**PreferencesMenu3**	
IP Phone – Phone Settings – GUI Menus	**PreferencesMenu4**	
IP Phone – Phone Settings – GUI Menus	**PreferencesMenu5**	
IP Phone – Phone Settings – GUI Menus	**PreferencesMenu6**	
Settings Menu		
IP Phone – Phone Settings – GUI Menus	**SettingsMenu1**	For example: `net;Network Settings,wifi;WiFi Settings,bt;Bluetooth Settings`
IP Phone – Phone Settings – GUI Menus	**SettingsMenu2**	
Product Information Menu		
IP Phone – Phone Settings – GUI Menus	**ProductInfoMenu1**	For example: `model;Model,obinumber;OBi Number,mac;MAC Address`
IP Phone – Phone Settings – GUI Menus	**ProductInfoMenu2**	
Main Menu		
IP Phone – Phone Settings – GUI Menus	**MainMenu1**	For example: `phone-book;Contacts,calls;Current Calls;call-history;Call History`
Net Services Menu		
IP Phone – Phone Settings – GUI Menus	**NetServicesMenu1**	For example: `bci;Anonymous Call,dnd,cfa,cfb,cfna,acd,bwanw, rmoff,simring`

Main Menu Item IDs

Main Menu Item ID	Default Caption	Description
phone-book	Contacts	Locally phone book
calls	Current Calls	List all the currently on-going calls
call-history	Call History	Locally stored call logs (for all calls on all services)
preferences	Preferences	Local phone feature preferences
settings	Settings	List all the system level settings
prod-info	Product Info	A list of data related to the product such as model and serial numbers, software and hardware versions.
netsrv	Net Services	Access the settings of many network provided features for each SP*n* service, *n* = 1 – 6.
netdir	Net Dir	Access the netwok directory associated with a specific SP*n* service or the LDAP service.
buddy	Buddy List	Access to the buddy list associated with a specific SP*n* service
directories	Directories	A container that lists the following options: - local phone-book (same as Contacts) - network directory for each SP*n* service, when available, *n* = 1 – 6 - LDAP Search (if configured) When only the local phone-book is available, this item behaves the same as the phone-book item.
messages	Messages	This option shows a list where each entry is the message or mailbox status of a SP*n* service, for *n* = 1 – 6, if enabled
all-preferences	Preferences	A container that lists the following options: - Phone (same as perferences item) - SP*n* Service (or its label), for *n* = 1 – 6, if enabled If none of SPn preferences are enabled, then the item behaves exactly as the legacy preferences item
call-histories	Call Histories	A container to access the locally stored call histories and the call history associated with each SP*n* service, *n* = 1 – 6.

Net Services Menu Item IDs

Net Services Menu Item ID	Default Caption	Description
acd	ACD Agent Sign On/Off	
bci	Anonymous Call	Turn Anonymous Call feature on/off
bwanw	BroadWorks Anywhere	
buddy	Buddy List	
ccs	Call Center Status	
cfa	Call Forward Always	
cfb	Call Forward Busy	
cfna	Call Forward No Answer	
rec	Call Recording Mode	
dnd	Do Not Disturb	
dispcode	Enter Call Disposition Code	
exec	Executive Call Filter	
xass	Executive Assistance	
hotel	Hoteling	
clog	Network Call History	
dir	Network Directory	
rmoff	Remote Office	
secClass	Security Class	
simring	Simultaneous Ring	

Preferences Menu Item IDs

Preferences	Default	Description

Menu Item ID	Caption	
language	Language	Select the Language of the displayed text on the phone screen
timeFormat	Time Format	Select the format of the time display on the phone screen
dateFormat	Date Format	Select the format of the date display on the phone screen
hpTime	Auto Home Page	Select the amount of idle time before automatically returning the phone screen to the Home screen
skin	Skin	Select the built-in GUI scheme (“skin”) to use
bgpic	Background Picture	Select an available background picture to use as wall paper
dring	Default Ringtone	Select an available ringtone as the default ringtone
dfont	Font	Select an available font as the default font for the displayed text
appCols	Home App Columns	Select the number of columns to display the GUI Main Menu in
packcalls	Pack Calls On Display	Enable/Disable packing calls on different services on the same screen of the display
sdim	Dim Screen	Enable/Disable dimming the screen (to the least settable Brightness level) after the set interval of no-key-presses-and-no-calls
sdimDelay	Dim Screen Delay	Interval of no-key-presses-and-no-calls in seconds before dimming the screen, if enabled
ssvr	Screen Saver	Enable/Disable the screen saver feature
ssvrDelay	Screen Saver Delay	Amount of idle time in seconds before starting the screen saver (if enabled)
ssvrType	Screen Saver Type	Select a screen save type
ssvrLock	Require Passcode On Wake Up	Select whether a passcode is required to exit from the screen saver
ssvrPass	Wake Up Passcode	Passcode to exit from screen saver
	Screen Saver Show Custom Contents	
brightness	Screen Brightness	Set the brightness of the LCD screen
audioDevice	Preferred Audio Device	Select whether to use speakerphone or a headset as the talk device when a headset is connected
headsetDevice	Preferred Headset Device	Select the preferred headset device when more than one is connected
ehs	Electronic Hook Switch	
dnd	Do Not Disturb	Enable/Disable Do Not Disturb Feature
dnr	Do Not Ring	Enable/Disable Do Not Ring Feature
cfa.enable	Call Forward	Enable/Disable Call Forwarding (all calls, unconditionally)
cwa	Call Waiting	Enable/Disable Call Waiting
bac	Block Anonymous Call	Enable/Disble Block Anonymous Caller Feature
bci	Anonymous Call	Enable/Disable Block Caller ID (a.k.a. Anonymous Call) Feature
aans	Auto Answer Page	Enable/Disable Auto-Answering incoming page
pg1	Join Page Group 1	Join Page Group 1
pg2	Join Page Group 2	Join Page Group 2
ringerVol	Ringer Volume	Ringer Volume
speakerVol	Speakerphone Volume	Speakerphone Audio Volume
micGain	Speakerphone Mic Gain	Speakerphone Mic Input Gain
handsetVol	Handset volume	Handset Audio Volume
handsetGain	Handset Mic Gain	Handset Mic Input Gain
headsetRJ9Vol	RJ9 Headset Voume	Audio Volume for the RJ9 Headset Device
headsetRJ9Gain	RJ9 Headset Mic Gain	Mic Input Gain for the RJ9 Headset Device
headset35mmVol	3.5mm Headset Volume	Audio Volume for the 3.5mm Headset Device
headset35mmGain	3.5mm Headset Mic Gain	Mic Input Gain for the 3.5mm Headset Device
headsetBTVol	BT Headset Volume	Audio Volume for the Bluetooth Headset Device
headsetBTGain	BT Headset Mic Gain	Mic Input Gain for the Bluetooth Headset Device
EqEnable	Equalizer	Enable the Equalizer
AecEnable	Acoustic Echo Cancellation	Enable the Acoustic Echo Canceller
edialer	Enhanced Dialer	Enhanced Dialer (show matching list of recently dialed numbers while dialing)
dlkt	Default Line Key Tab	Default Line Key Tab (the Line Key Tab to fallback to after a configurable duration of idle time)

Settings Menu Item IDs

Settings Menu Item ID	Default Caption	Description
net	Network	Show current IP address and launch Network Settings Menu
wifi	WiFi	Show WiFi connection status and launch Wifi Settings Menu
bt	Bluetooth	Show Bluetooth connection status and launch Bluetooth Settings Menu
obiline	OBiLine	Set up OBiLine (USB FXO Port Adapter)
flash	Storage	View, read, or copy contents in the internal user data storage or external USB thumb drives
ringfile	Ringtones	Set up the default ring tone per SP*n* Service
pkeys	Programmable Keys	Launch Programmable Keys Settings Menu
lkeys	Line Keys	Launch Line Keys Settings Menu
sidecar1	Side Car 1	Show Side Car 1 connection status and launch Side Car 1 Settings Menu. Not available on OBi1022 and OBi2000 Series
sidecar2	Side Car 2	Show Side Car 2 connection status and launch Side Car 2 Settings Menu. Not available on OBi1022 and OBi2000 Series
voice	Voice Services	Launch Voice Services Settings Menu
Sd99	Speed Dials 99	Set up 99 speed dial numbers
clrcache	Clear Data Cache	Clear the downloaded data cache
admin	Device Administration	Show Administrative Settings Menu
login	Unlock Admin Settings	Login or logut administrative mode

Product Info Menu Item IDs

Product Info Menu Item ID	Default Caption	Description
model	Model	Model Number
obinumber	OBi Nuber	OBi Number
mac	MAC Address	(Ethernet) MAC Address
wfmac	WiFi MAC Address	WiFi MAC Address
serial	Serial Number	Serial Number
swver	Software Version	Software Version
hwver	Hardware Version	Hardware Version
ztinfo	Customization Status	ZT Customization Status
uptime	Up Time	UP time since last bootup

Cache Control of Downloaded (Temporary) Data

The following data items can be downloaded and cached by the phone at run time (in DRAM):

- Ring tone file downloaded from the URL specified in ***User Preferences***::**DefaultRingtone**
- Picture file downloaded from the URL specified in ***User Preferences***::**BackgroundPicture**
- Caller ID Picture files downloaded from URL extracted from Call-Info header of SIP Messages
- Picture files downloaded from URL in *src* attribute of <img> elements in rendered <ScreenItem>
- Var-Tree XML files downloaded from URL in *value* attribute of <setvar> elements in rendered <ScreenItem>
- Icon, Wave or MP3 files downloaded when rendering OBiPhone XML Apps

The expiration time of each individually cached data file is taken from the HTTP/Cache-Control header in the 200 response when the file is originally recevied from the server. If the Cache-Control header is present in the 200 response with the *no-cache* flag specified, the downloaded data file is not cached. Otherwise, if the Cache-Control header is present with the *max-age* attribute specified, the data file is cached for the number of seconds as specified in the *max-*

age attribute. Otherwise, the data file is cached indefinitely until one of the following happens that clears ALL the cached data:

- Phone is power cycled, or after a full reboot (such as after a firmware update)
- User selects "Clear Data Cache" option on the Settings Menu from the phone GUI
- User enables the ***Phone Settings***::**ClearDownloadedDataCache** option on the web page and submits the change
- Administrator provisions the phone with the value of the parameter **VoiceService.1.Phone.ClearDownloadedDataCache** set to true

Phone Customization Data

In addition to the configurable aspects of the GUI discussed above, some of the data that drives the GUI can be customized by the administrator of the phones. These data are collectively called Phone Customization Data which include the following data types:

Customization Data Type	Description	Internal Data Storage Path
Startup Splash Screen	The background picture shown on screen when the phone is starting up. The data file must be in 16bit RGB565 raw format and the filename must be logo.raw	/backgnd/logo.raw
Background Pictures (Wall Paper)	The pictures that can be chosen by the user to show as wall paper during normal phone operation. The data files can be JPEG, PNG, BMP, or GIF	/backgnd/
Dictionary	The translation to be used for various items shown on the phone screen	/dict/
Text Fonts	The true-type font files for rendering text to display on the screen	/fonts/
Ring Tones	Ring tone wave files. It must be in 16-bit monon linear PCM wave format sampled at 16 kHz	/ringtones/

Startup Splash Screen

During cold boot, the phone looks for a custom splash screen *logo.raw* in the internal folder */backgnd/* (the same folder where custom background pictures are located).

logo.raw must have a dimension of 480x272, and must be stored in 16 bits per pixel RGB565 format. An example of converting a PNG file to a *logo.raw* on a Linux system is shown here:

```
avconv -vcodec png -i mylogo.png -vcodec rawvideo -f rawvideo -pix_fmt rgb565 logo.raw
```

The file size of *logo.raw* should always be 261120 bytes. A custom logo.raw can only be uploaded into the phone using the customization data package upload method.

Background Pictures

Custom background pictures are stored in the internal folder */backgnd/.* Each background picture should have the pixel dimension 480Wx272H. JPEG, PNG, BMP, and GIF files are the acceptable file formats. All the pictures stored in this folder will be made available to be selected by the user as background picture (i.e. wall paper) on the phone screen.

There is a “Background Picture” entry under the “Preference” menu of the phone, with which the end user can browse and select among the available background pictures (built-in and custom). Each of the three built-in “Skins” has its own default background pictures.

End-users may upload their own background pictures by copying them from USB flash drive using the Storage Explorer option under the phone’s Settings menu. The destination folder to copy to is */backgnd/.*

Text Fonts Customization

Fonts are stored in the internal folder */fonts/*. Custom font files can only be installed on the phone using the data package upload method. Acceptable font file formats are TTF and OTF. In order to choose an installed font, you must also add a font-name-to-font-file for each installed font in the “font.map” file that is also stored in the same folder. “font.map” maps the font nameto the actual font file. The set of font names listed in “font.map” is also presented to the user as a list of selections under “Preferences/Font” in the phone GUI. On the web page the DefaultFont parameter is just a string and should match one of the font names (not file names) listed in “font.map”.

Below is the default contents of “font.map” which lists all the factory-installed fonts that come with the phone.

```
#
# font.map
#
# The left column is a font name that a user can select from
# the Preferences menu. The right column is the corresponding
# physical font file installed on the phone.
# A valid font file must either a ttf or otf file.
#
# You can use any number of white spaces as delimiters between the 2 columns.
#
aleo                 Aleo-Light.otf
aleo-bold            Aleo-Bold.otf
aleo-italic          Aleo-LightItalic.otf
aleo-bold-italic     Aleo-BoldItalic.otf
banksia              Banksia-Regular.otf
banksia-bold         Banksia-Bold.otf
droidsans            DroidSans.ttf
droidsans-bold       DroidSans-Bold.ttf
libre-caslon         LibreCaslonText-Regular.ttf
libre-caslon-bold    LibreCaslonText-Bold.ttf
libre-caslon-italic  LibreCaslonText-Italic.ttf
opensans             OpenSans-Regular.ttf
opensans-bold        OpenSans-Bold.ttf
opensans-italic      OpenSans-Italic.ttf
opensans-bold-italic OpenSans-BoldItalic.ttf
quattrocento         Quattrocento-Regular.ttf
quattrocento-bold    Quattrocento-Bold.ttf
terminal-dosis       TerminalDosis-Regular.ttf
terminal-dosis-bold  TerminalDosis-Bold.ttf
```

In addtiion, there are four more built-in fonts:

```
ptsans,
ptsans-bold
ptsans-italic,
ptsans-bold-italic,
```

Font	Sample
aleo	abcdefghijklmnopqrstuvwxyz ABCDEFGHIJKLMNOPQRSTUVWXYZ 1234567890.:,; ' " (!?) +-*/=
aleo-bold	**abcdefghijklmnopqrstuvwxyz ABCDEFGHIJKLMNOPQRSTUVWXYZ 1234567890.:,; ' " (!?) +-*/=**
aleo-bold-italic	***abcdefghijklmnopqrstuvwxyz ABCDEFGHIJKLMNOPQRSTUVWXYZ 1234567890.:,; ' " (!?) +-*/=***

aleo-italic	*abcdefghijklmnopqrstuvwxyz ABCDEFGHIJKLMNOPQRSTUVWXYZ 1234567890.:,; ' " (!?) +-*/=*
banksia	abcdefghijklmnopqrstuvwxyz ABCDEFGHIJKLMNOPQRSTUVWXYZ 1234567890.:,; ' " (!?) +-*/=
banksia-bold	**abcdefghijklmnopqrstuvwxyz ABCDEFGHIJKLMNOPQRSTUVWXYZ 1234567890.:,; ' " (!?) +-*/=**
droidsans	abcdefghijklmnopqrstuvwxyz ABCDEFGHIJKLMNOPQRSTUVWXYZ 1234567890.:,; ' " (!?) +-*/=
droidsans-bold	**abcdefghijklmnopqrstuvwxyz ABCDEFGHIJKLMNOPQRSTUVWXYZ 1234567890.:,; ' " (!?) +-*/=**
libre-caslon	abcdefghijklmnopqrstuvwxyz ABCDEFGHIJKLMNOPQRSTUVWXYZ 1234567890.:,; ' " (!?) +-*/=
libre-caslon-italic	*abcdefghijklmnopqrstuvwxyz ABCDEFGHIJKLMNOPQRSTUVWXYZ 1234567890.:,; ' " (!?) +-*/=*
libre-caslon-bold	**abcdefghijklmnopqrstuvwxyz ABCDEFGHIJKLMNOPQRSTUVWXYZ 1234567890.:,; ' " (!?) +-*/=**
opensans	abcdefghijklmnopqrstuvwxyz ABCDEFGHIJKLMNOPQRSTUVWXYZ 1234567890.:,; ' " (!?) +-*/=
opensans-italic	*abcdefghijklmnopqrstuvwxyz ABCDEFGHIJKLMNOPQRSTUVWXYZ 1234567890.:,; ' " (!?) +-*/=*
opensans-bold	**abcdefghijklmnopqrstuvwxyz ABCDEFGHIJKLMNOPQRSTUVWXYZ 1234567890.:,; ' " (!?) +-*/=**
opensans-bold-italic	***abcdefghijklmnopqrstuvwxyz ABCDEFGHIJKLMNOPQRSTUVWXYZ 1234567890.:,; ' " (!?) +-*/=***
quattrocento	abcdefghijklmnopqrstuvwxyz ABCDEFGHIJKLMNOPQRSTUVWXYZ 1234567890.:,; ' " (!?) +-*/=
quattrocento -bold	**abcdefghijklmnopqrstuvwxyz ABCDEFGHIJKLMNOPQRSTUVWXYZ 1234567890.:,; ' " (!?) +-*/=**
terminal-dosis	abcdefghijklmnopqrstuvwxyz ABCDEFGHIJKLMNOPQRSTUVWXYZ 1234567890.:,; ' " (!?) +-*/=
terminal-dosis-bold	**abcdefghijklmnopqrstuvwxyz ABCDEFGHIJKLMNOPQRSTUVWXYZ 1234567890.:,; ' " (!?) +-*/=**
ptsans	abcdefghijklmnopqrstuvwxyz ABCDEFGHIJKLMNOPQRSTUVWXYZ 1234567890.:,; ' " (!?) +-*/=
ptsans-italic	*abcdefghijklmnopqrstuvwxyz ABCDEFGHIJKLMNOPQRSTUVWXYZ 1234567890.:,; ' " (!?) +-*/=*
ptsans-bold	**abcdefghijklmnopqrstuvwxyz ABCDEFGHIJKLMNOPQRSTUVWXYZ 1234567890.:,; ' " (!?) +-*/=**
ptsans-bold-italic	***abcdefghijklmnopqrstuvwxyz ABCDEFGHIJKLMNOPQRSTUVWXYZ 1234567890.:,; ' " (!?) +-*/=***

DefaultFont parameter takes only the font family; the style suffix is automatically applied by the phone. Hence the following built-in font families can be used:

- ptsans
- aleo
- banksia
- droidsans
- libre-caslon
- opensans
- quattrocento
- terminal-dosis

Notes:

- When using the Tomas skin, however, the font is fixed. The configured DefaultFont value will not take effect
- When using a language other than English, make sure to use a Font family that includes all the characters in the chosen language

Language Customization with Dictionary Files

Language customization allows you to change the language of the displayed text shown on the phone GUI. The customization is done by installing a dictionary file for each selectable language. Dictionary files are XML files stored in the internal folder */dict/* and must be UTF-8 encoded. Custom dictionary files may only be installed on the phone using the data package upload method. By default the phone comes with three built-in dictionary files: English-US, English-UK, and Spanish. User can set select the language under Preferences on the GUI.

Below is the full Spanish dictionary file that is shipped with the phone. The name of the language is specified in the lang attribute of the root element <dictionary> and is also the name shown to the user in the Perference for language selection.

```
<!-- Each dictionary item has a key/value pair, a mode attribute and a namespace.
    When applying the dictionary, we match the key against the source phrase.
    If mode="i", the comparison is case-insensitive; otherwise the comparison
    is case-sensitive. The namespace ns must be matched also.
-->
<dictionary lang="Español">
  <!-- App names -->
  <item key="Contacts" value="Contactos"/>
  <item key="Current Calls" value="Llamadas"/>
  <item key="Call History" value="Historial"/>
  <item key="Preferences" value="Preferencias"/>
  <item key="Settings" value="Ajustes"/>
  <item key="Product Info" value="Producto"/>
  <item key="Buddy List" value="Amistades"/>
  <item key="Net Services" value="Red Servicios"/>

  <!-- Line key function name -->
  <item key="AANS" value="Respuesta Automática" mode="i"/>
  <item key="acd" value="Distr. automático de llamadas" mode="i"/>
  <item key="BXFER" value="Transferencia Ciega" mode="i"/>
  <item key="BAC" value="Blq. Llamadas Anónimas" mode="i"/>
  <item key="BCI" value="Blq. ID de Llamadas" mode="i"/>
  <item key="BLF" value="Monitor de teléfono" mode="i"/>
  <item key="call" value="Llamada" mode="i"/>
  <item key="CFA" value="Desvío de Llamadas" mode="i"/>
  <item key="CWA" value="Llamada en Espera" mode="i"/>
  <item key="CONF" value="Unir a Conferencia" mode="i"/>
  <item key="disp-code" value="Código Disposición" mode="i"/>
  <item key="DND" value="No Molestar" mode="i"/>
  <item key="DNR" value="No Suena" mode="i"/>
  <item key="HOLD" value="En Espera" mode="i"/>
  <item key="hoteling" value="Hoteling" mode="i"/>
  <item key="acd" value="ACD Agent State" mode="i"/>
```

```
<item key="exe" value="Exec Call Filter" mode="i"/>
<item key="xass" value="Exec Assistant" mode="i"/>
<item key="sec"  value="Security Class" mode="i"/>
<item key="LINE" value="Monitor Linea" mode="i"/>
<item key="lkpg" value="Siguiente Tabla" mode="i"/>
<item key="MWI" value="Mensajes" mode="i"/>
<item key="pg1" value="Grupo de Páginas 1"/>
<item key="pg2" value="Grupo de Páginas 2"/>
<item key="SPD" value="Marcado Rápido" mode="i"/>
<item key="XFER" value="Transferencia" mode="i"/>
<item key="pres" value="Presencia" mode="i"/>
<item key="cpm" value="Monitor de Aparcamiento de Llamada" mode="i"/>

<!-- Product Info entries -->
<item key="Model" value="Modelo" mode="i"/>
<item key="OBi Number" value="Número OBi" mode="i"/>
<item key="MAC Address" value="Dirección MAC" mode="i"/>
<item key="WiFi MAC Address" value="Dirección MAC de WiFi" mode="i"/>
<item key="Serial Number" value="Número Serial" mode="i"/>
<item key="Software Version" value="Versión del Software" mode="i"/>
<item key="Hardware Version" value="Versión del Hardware" mode="i"/>
<item key="Customization Status" value="Estado de Personalización" mode="i"/>
<item key="Up Time" value="Tiempo de Actividad" mode="i"/>

<!-- User Preferences entries -->
<item key="Language" value="Lenguaje" mode="i"/>
<item key="Skin" value="Contraste" mode="i"/>
<item key="Background Picture" value="Imagen de Fondo" mode="i"/>
<item key="Default Ringtone" value="Timbre Predeterminado" mode="i"/>
<item key="Default Font" value="Fuente Predeterminada" mode="i"/>
<item key="Screen Saver" value="Protector de Pantalla" mode="i"/>
<item key="Screen Saver Delay" value="Protector de Pantalla de retardo" mode="i"/>
<item key="Screen Saver Type" value="Tipo de Protector de Pantalla" mode="i"/>
<item key="Screen Brightness" value="Brillo de la Pantalla" mode="i"/>
<item key="Preferred Audio Device" value="Dispositivo de Audio Preferido" mode="i"/>
<item key="Preferred Headset Device" value="Auriculares Preferido" mode="i"/>
<item key="Do Not Disturb" value="No Molestar" mode="i"/>
<item key="Do Not Ring" value="No Suena" mode="i"/>
<item key="Call Forward" value="Desvío de Llamadas" mode="i"/>
<item key="Call Waiting" value="Llamada en Espera" mode="i"/>
<item key="Block Anonymous Call" value="Bloq. Llamadas Anónimas" mode="i"/>
<item key="Anonymous Call" value="Llamadas Anónimas" mode="i"/>
<item key="Auto Answer Page" value="Pagina de Respuesta Automatica" mode="i"/>
<item key="Join Page Group" value="Únete Grupo de Páginas" mode="i"/>
<item key="Ringer Volume" value="Volumen del Timbre" mode="i"/>
<item key="Speakerphone Volume" value="Volumen de Altavoz" mode="i"/>
<item key="Speakerphone Mic Gain" value="Ganancia Mic Altavoz" mode="i"/>
<item key="Handset Volume" value="Volumen de Auriculares" mode="i"/>
<item key="Handset Mic Gain" value="Ganancia Mic Auriculares" mode="i"/>
<item key="RJ9 Headset Volume" value="RJ9 Volumen de Auricular" mode="i"/>
<item key="RJ9 Headset Mic Gain" value="RJ9 Ganancia Mic Auricular" mode="i"/>
<item key="3.5mm Headset Volume" value="3.5mm Volumen de Auricular" mode="i"/>
<item key="3.5mm Headset Mic Gain" value="3.5mm Ganancia Mic Auricular" mode="i"/>
<item key="BT Headset Volume" value="BT Volumen de Auricular" mode="i"/>
<item key="BT Headset Mic Gain" value="BT Ganancia Mic Auricular" mode="i"/>
<item key="Equalizer" value="Equalizador" mode="i"/>
<item key="AEC" value="Cancelación de Eco Acústico" mode="i"/>
<item key="Home App Columns" value="Columnas de Aplicación Home" mode="i"/>
<!-- Screen Saver Type -->
<item key="Slide Show" value="Pantalla de Visualizacion" mode="i"/>
<item key="Turn Off Display" value="Desactivar la Visualización" mode="i"/>
<!-- Preferred Audio Device -->
<item key="Speaker" value="Altavoz" mode="i"/>
<item key="Handset" value="Auriculares" mode="i"/>
<item key="Headset" value="Auricular" mode="i"/>
<!-- Preferred Headset Device -->
<item key="RJ9 Headset" value="RJ9 Auricular" mode="i"/>
<item key="3.5mm Headset" value="3.5mm Auricular" mode="i"/>
<item key="BT Headset" value="BT Auricular" mode="i"/>

<!-- Call Forward -->
<item key="Destination Number" value="Número de Destino" mode="i"/>
<item key="Voice Service" value="Servicio de Voz" mode="i"/>
```

```
<!-- Phone Book -->
<item key="Phone Book" value="Agenda" mode="i"/>
<item key="Fields" value="Campos" mode="i"/>
<item key="Name" value="Nombre" mode="i"/>
<item key="First Name" value="Primer Nombre" mode="i"/>
<item key="Last Name" value="Apellidos" mode="i"/>
<item key="Number" value="Número" mode="i"/>
<item key="Mobile Number" value="Número Movil" mode="i"/>
<item key="Office Number" value="Número de Oficina" mode="i"/>
<item key="Home Number" value="Numero de Casa" mode="i"/>
<item key="Service" value="Servicio" mode="i"/>
<item key="Picture" value="Foto" mode="i"/>
<item key="Ringtone" value="Timbre" mode="i"/>
<item key="Email" value="Email" mode="i"/>
<item key="Company" value="Empresa" mode="i"/>
<item key="Group" value="Grupo" mode="i"/>
<item key="Groups" value="Groupos" mode="i"/>
<item key="Co-Workers" value="Compañeros de Trabajo" mode="i"/>
<item key="Customers" value="Clientes" mode="i"/>
<item key="Family" value="Familia" mode="i"/>
<item key="Friends" value="Amigos" mode="i"/>
<item key="No Groups Defined" value="Grupo No Definido" mode="i"/>
<item key="Speed Dial" value="Marcación Rápida" mode="i"/>
<item key="Phone Book Filter" value="Filtro de Agenda" mode="i"/>

<!-- Call History -->
<item key="Missed Calls" value="Llamadas Perdidas" mode="i"/>
<item key="Received Calls" value="Llamadas Recibidas" mode="i"/>
<item key="Outgoing Calls" value="Llamadas Realizadas" mode="i"/>
<item key="All Calls" value="Todas las Llamadas" mode="i"/>
<item key="List empty" value="Lista Vacia" mode="i"/>

<!-- Buddy List -->
<item key="My Status" value="Mi Estatus" mode="i"/>
<item key="My Presence" value="Mi Presencia" mode="i"/>
<!-- Presence status -->
<item key="unknown" value="Desconocido" mode="i"/>
<item key="offline" value="Fuera de Linea" mode="i"/>
<item key="online" value="En Linea" mode="i"/>
<item key="away" value="Fuera de Oficina" mode="i"/>
<item key="xa" value="Fuera de Oficina Extendida" mode="i"/>
<item key="dnd" value="No Molestar" mode="i"/>
<item key="Online (Available)" value="En Linea (Disponible)" mode="i"/>
<item key="Offline (Invisible)" value="Fuera de Linea (Invisible)" mode="i"/>
<item key="Extended Away" value="Fuera de Oficina Extendida" mode="i"/>
<item key="DND (Do Not Disturb)" value="DND (No Molestar)" mode="i"/>

<!-- Network Directory -->
<item key="Enterprise" value="Empresa" mode="i"/>
<item key="Group Common" value="Grupo en Comun" mode="i"/>
<item key="Enterprise Common" value="Empresa en Comun" mode="i"/>
<item key="Personal" value="Personal" mode="i"/>

<!-- Settings -->
<item key="Network" value="Reseteo de Fabrica" mode="i"/>
<item key="Product Information" value="Informacion Producto" mode="i"/>
<item key="WiFi" value="WiFi" mode="i"/>
<item key="Bluetooth" value="Bluetooth" mode="i"/>
<item key="Storage" value="Almacenamiento" mode="i"/>
<item key="User Preferences" value="Preferencias Usuario" mode="i"/>
<item key="Programmable Keys" value="Teclas Programables" mode="i"/>
<item key="Line Keys" value="Teclas de Linea" mode="i"/>
<item key="Side Car 1" value="Consola de Operadora 1" mode="i"/>
<item key="Side Car 2" value="Consola de Operadora 2" mode="i"/>
<item key="Voice Services" value="Servicios de Voz" mode="i"/>
<item key="Device Admin" value="Administracion Dispositivo" mode="i"/>
<item key="Lock Admin Settings" value="Bloqueo Config. del Admin." mode="i"/>
<item key="Unlock Admin Settings" value="Desbloquear Ajustes del Admin." mode="i"/>
<!-- Network -->
<item key="Network Settings" value="Ajustes de la Red" mode="i"/>
<item key="Addressing Type" value="Addressing Type" mode="i"/>
<item key="IP Address" value="Direccion IP" mode="i"/>
```

```
<item key="Subnet Mask" value="Mascara Subred" mode="i"/>
<item key="Default Gateway" value="Direccion Gateway" mode="i"/>
<item key="DNS Server" value="Servidor DNS" mode="i"/>
<item key="DNS Query Order" value="Orden de Consulta DNS" mode="i"/>
<item key="DNS Query Delay" value="Retrazo de Consulta DNS" mode="i"/>
<item key="PPPoE" value="PPPoE" mode="i"/>
<item key="AC Name" value="Nombre AC" mode="i"/>
<item key="Service Name" value="Nombre de Servicio" mode="i"/>
<item key="Password" value="Contraseña" mode="i"/>
<item key="VLAN" value="VLAN" mode="i"/>
<item key="LLDP-MED" value="LLDP-MED" mode="i"/>
<item key="ID" value="ID" mode="i"/>
<item key="Priority" value="Prioridad" mode="i"/>
<item key="NTP Server" value="Servidor NTP" mode="i"/>
<item key="Local Time Zone" value="Zona Horaria Local" mode="i"/>
<item key="Daylight Saving Time" value="Horario de Verano" mode="i"/>
<!-- WiFi -->
<item key="WiFi Setup" value="Configuracion WiFi" mode="i"/>
<item key="OBiWiFi Setup Mode" value="Modo de configuración OBiWiFi" mode="i"/>
<item key="Security" value="Seguridad" mode="i"/>
<item key="Signal Strength" value="Intensidad de Señal" mode="i"/>
<item key="MAC Address" value="Direccion MAC" mode="i"/>
<item key="0" value="" ns="wfs"/>
<item key="1" value="Débil" ns="wfs"/>
<item key="2" value="Pobre" ns="wfs"/>
<item key="3" value="Pasable" ns="wfs"/>
<item key="4" value="Bueno" ns="wfs"/>
<item key="5" value="Excelente" ns="wfs"/>
<!-- Bluetooth -->
<item key="Bluetooth Setup" value="Configuracion Bluetooth" mode="i"/>
<item key="Pairing Mode" value="Modo de Asociación" mode="i"/>
<item key="enabled" value="Asociar con un Audífono" ns="bt" mode="i"/>
<item key="disabled" value="Asociar con el Teléfono Móvil" ns="bt" mode="i"/>
<item key="Discoverable" value="Reconocible" mode="i"/>
<!-- Storage -->
<item key="Device" value="Dispositivo" mode="i"/>
<item key="File System Type" value="Tipo de Sistema de Archivo" mode="i"/>
<item key="Capacity" value="Capacidad" mode="i"/>
<item key="Internal" value="Interno" mode="i"/>
<item key="USB storage 1" value="USB Flash Drive 1" mode="i"/>
<item key="USB storage 2" value="USB Flash Drive 2" mode="i"/>
<!-- Voice Services -->
<!--
 Highly technical entries under Voice Services are not translated.
-->
<!-- Admin Settings -->
<item key="Web Server Port" value="Puerto del Servidor Web" mode="i"/>
<item key="Web Admin Password" value="Contraseña del Administrador" mode="i"/>
<item key="Web User Password" value="Contraseña del Usuario" mode="i"/>
<item key="Syslog Server" value="Servidor Syslog" mode="i"/>
<item key="ITSP Provisioning Method" value="Método Aprovisionamiento ITSP" mode="i"/>
<item key="ITSP Provisioning Interval" value="Aprovisionamiento Interno ITSP" mode="i"/>
<item key="ITSP Provisioning Config URL" value="Aprovisionamiento ITSP Config URL" mode="i"/>
<item key="Auto Firmware Update Method" value="Metodos de Actualizacion Auto Firmware" mode="i"/>
<item key="Auto Firmware Update Interval" value="Actualizacion Interno Auto Firmware" mode="i"/>
<item key="Auto Firmware Update URL" value="Actualizacion Auto Firmware URL" mode="i"/>
<!-- Provisioning and FW Upgrade Method -->
<item key="System Start" value="Inicio del Sistema" mode="i"/>
<item key="Periodically" value="Periodicamente" mode="i"/>
<!-- Dial App -->
<item key="Enter Number" value="Número de Entrada" mode="i"/>
<item key="Transfer Target" value="Destino de la Transferencia" mode="i"/>
<item key="Conference Target" value="Objetivo Conferencia" mode="i"/>

<!-- Soft Keys -->
<item key="Edit" value="Editar" mode="i"/>
<item key="Reboot" value="Reiniciar" mode="i"/>
<item key="End" value="Final" mode="i"/>
<item key="Hold" value="Espera" mode="i"/>
<item key="Resume" value="Resumen" mode="i"/>
<item key="Add to Conf" value="Unir a Conferencia" mode="i"/>
<item key="Conference" value="Conferencia" mode="i"/>
<item key="Park" value="Aparcar" mode="i"/>
```

```
<item key="Blind Transfer" value="Transferencia Ciega" mode="i"/>
<item key="Dispose Code" value="Deseche Código" mode="i"/>
<item key="Escalate" value="Escalar" mode="i"/>
<item key="Trace" value="Trazar" mode="i"/>
<item key="Rec.Start" value="Iniciar Grabacion" mode="i"/>
<item key="Rec.Stop" value="Parar Grabacion" mode="i"/>
<item key="Rec.Pause" value="Pausar Grabacion" mode="i"/>
<item key="Rec.Resume" value="Resumir Grabacion" mode="i"/>
<item key="Private Hold" value="Espera Privada" mode="i"/>
<item key="Answer" value="Contestar" mode="i"/>
<item key="Reject" value="Rechazar" mode="i"/>
<item key="Redial" value="Rellamar" mode="i"/>
<item key="Missed" value="Perdida" mode="i"/>
<item key="Dial" value="Marcar" mode="i"/>
<item key="Lines" value="Lineas" mode="i"/>
<item key="Switch Mode" value="Cambio Modo" mode="i"/>
<item key="Ph.Book" value="Agenda" mode="i"/>
<item key="Switch Line" value="Cambio Linea" mode="i"/>
<item key="Refresh All" value="Reiniciar Todo" mode="i"/>
<item key="+Buddy" value="Agregar Amigo" mode="i"/>
<item key="Search" value="Busqueda" mode="i"/>
<item key="Refresh" value="Reiniciar" mode="i"/>
<item key="+" value="Agregar" mode="i"/>
<item key="Add" value="Agregar" mode="i"/>
<item key="Remove" value="Remover" mode="i"/>
<item key="Clear List" value="Limpiar Lista" mode="i"/>
<item key="Edit Dial" value="Aditar Marcador" mode="i"/>
<item key="Save" value="Guardar" mode="i"/>
<item key="MyPresence" value="Mi Presencia" mode="i"/>
<item key="MyStatus" value="Mi Estatus" mode="i"/>
<item key="Filter" value="Filtrar" mode="i"/>
<item key="Select" value="Seleccionar" mode="i"/>
<item key="New" value="Nuevo" mode="i"/>
<item key="Clear" value="Limpiar" mode="i"/>
<item key="Sort" value="Ordenar" mode="i"/>
<item key="New Entry" value="Nueva Entrada" mode="i"/>
<item key="Group Filter" value="Filtrar Grupo" mode="i"/>
<item key="Import" value="Importar" mode="i"/>
<item key="Export" value="Exportar" mode="i"/>
<item key="Sync" value="Sincronizar" mode="i"/>
<item key="New Group" value="Nuevo Grupo" mode="i"/>
<item key="Uncheck All" value="No chequear Nada" mode="i"/>
<item key="Check All" value="Chequear Todo" mode="i"/>
<item key="Factory Reset" value="Reseteo de Fabrica" mode="i"/>
<item key="FW Update" value=" Actializar FW" mode="i"/>
<item key="Pick Up" value="Contestar" mode="i"/>
<item key="Barge In" value="Interrumpir" mode="i"/>
<item key="Monitor" value="Monitorear" mode="i"/>
<item key="Sign Off" value="Cerrar Sesión" mode="i"/>
<item key="Wrap Up" value="Envolver" mode="i"/>
<item key="Scan" value="Escanear" mode="i"/>
<item key="Transfer Now" value="Transferir Ahora" mode="i"/>
<item key="Conference Now" value="Conferencia Ahora" mode="i"/>
<item key="Play" value="Seguir" mode="i"/>
<item key="Stop" value="Parar" mode="i"/>
<item key="Connect" value="Conectar" mode="i"/>
<item key="Rescan" value="Re-escanear" mode="i"/>
<item key="Disconnect" value="Desconectar" mode="i"/>
<item key="Forget" value="Olvidar" mode="i"/>
<item key="Reconnect" value="Reconectar" mode="i"/>
<item key="Explore" value="Explorar" mode="i"/>
<item key="Eject" value="Ejecutar" mode="i"/>
<item key="Mount" value="Montura" mode="i"/>
<item key="Back" value="Regresar" mode="i"/>
<item key="Copy" value="Copiar" mode="i"/>
<item key="Cancel" value="Cancelar" mode="i"/>
<item key="Paste" value="Pasar" mode="i"/>
<item key="New Folder" value="Nuevo Folder" mode="i"/>
<item key="Close" value="Cerrar" mode="i"/>
<item key="OK" value="OK" mode="i"/>
<item key="Replace" value="Reemplazar" mode="i"/>
<item key="Add As New" value="Agragar Como Nuevo" mode="i"/>
```

```
<item key="&gt;" value="&gt;"/>
<item key="&lt;" value="&lt;"/>
<item key="Backspace" value="Retroceso" mode="i"/>
<item key="Devices" value="Dispositivos" mode="i"/>

<!-- Misc -->
<item key="Not Found" value="Extraviado" mode="i"/>
<item key="No Entries" value="No Entradas" mode="i"/>
<item key="Unassigned" value="No Asignado" mode="i"/>
<item key="Enable" value="Activar" mode="i"/>
<item key="enabled" value="Activado" mode="i"/>
<item key="disabled" value="Desactivado" mode="i"/>
<item key="Not Configured" value="No Configurado" mode="i"/>
<item key="default" value="Por Defecto" mode="i"/>
<item key="On" value="Encendido" mode="i"/>
<item key="Off" value="Apagado" mode="i"/>
<item key="Non-linear" value="No Linear" mode="i"/>
<item key="Status" value="Estatus" mode="i"/>
<item key="outgoing" value="A" ns="ct"/>
<item key="received" value="De" ns="ct"/>
<item key="up" value="Arriba" mode="i"/>
<item key="down" value="Abajo" mode="i"/>
<item key="nocfg" value="No Configurado" mode="i"/>
<item key="holding" value="Esperando" mode="i" ns="cs"/>
<item key="peerring" value="Timbrando" mode="i" ns="cs"/>
<item key="proceeding" value="Timbrando" mode="i" ns="cs"/>
<item key="peerring2" value="Timbrando (Espera)" mode="i" ns="cs"/>
<item key="peerring3" value="Timbrando (Conf)" mode="i" ns="cs"/>
<item key="dialtone" value="Tono de Marcación" mode="i" ns="cs"/>
<item key="dialing" value="Marcando" mode="i" ns="cs"/>
<item key="trying" value="Tratando" mode="i" ns="cs"/>
<item key="connected" value="Conectado" mode="i" ns="cs"/>
<item key="connected-HD" value="HD Conectado" mode="i" ns="cs"/>
<item key="ended" value="Llamada Finalizada" mode="i" ns="cs"/>
<item key="ring" value="Timbrando" mode="i" ns="cs"/>
<item key="OBiTALK" value="OBiTALK" mode="i"/>
<item key="Auto" value="" mode="i"/>
<item key="Auto: " value="" mode="i"/>
<item key="OBiTALK: " value="OBiTALK: " mode="i"/>
<item key="Not Available" value="No Disponible" mode="i"/>
<item key="AA1: " value="OBi Asistente: " mode="i"/>
<item key="AA2: " value="IVR: " mode="i"/>
<item key="seized" value="Incautado" ns="cs"/>
<item key="held_private" value="Held Privado" ns="cs"/>
<item key="held" value="Held" ns="cs"/>
<item key="parked" value="Llamar Estacionado" ns="cs"/>

<!-- Misc line key data -->
<item key="error" value="error" mode="i"/>
<item key="Service Not Ready" value="Servicio no está listo" mode="i"/>
<item key="Service Subs. Error" value="Error suscripción de servicio" mode="i"/>
<item key="BLF Subs. Error" value="BLF error suscripción" mode="i"/>
<item key="Hoteling Subs. Error" value="Hoteling error suscripción" mode="i"/>
<item key="Guest" value="Invitado" mode="i"/>
<item key="No Guest" value="Los huéspedes no" mode="i"/>

<!-- ACD State -->
<item key="available" value="Disponible" mode="i"/>
<item key="unavailable" value="No Disponible" mode="i"/>
<item key="wrappingup" value="Terminando" mode="i"/>
<item key="signedoff" value="Cerrar Sesión" mode="i"/>

<!-- Input App Titles -->
<item key="New Status" value="Nuevo Estatus" mode="i"/>
<item key="Edit Status" value="Editar Estatus" mode="i"/>
<item key="New Group Name" value="Nuevo Nombre de Grupo" mode="i"/>
<item key="Enter" value="Entrar" mode="i"/>
<item key="Blind Transfer Target" value="Objetivo Transferencia ciega" mode="i"/>
<item key="Supervisor Extension" value="Supervisor de Extensión" mode="i"/>
<item key="Disposition Code" value="Disposicion de Codigo" mode="i"/>
<item key="Forward All Number" value="Remitir Todo Número" mode="i"/>
<item key="Unavailable Reason Code" value="No disponible Código de Motivo" mode="i"/>
<item key="Enter Query" value="Entrar Consulta" mode="i"/>
```

```
<item key="Hoteling Passcode" value="Codigo de Contraseña Hoteling" mode="i"/>
<item key="Hoteling Guest Extension" value="Extension Huesped Hoteling" mode="i"/>
<item key="Enter Admin Password" value="Ingrese Contraseña del Administrador" mode="i"/>
<item key="Enter Folder Name" value="Ingrese Nombre del Archivo" mode="i"/>
<item key="Exec Assist Divert Number" value="Exec Asist Numero de Desvio" mode="i"/>
<!-- Call related alerts -->
<item key="No Active Calls" value="No hay Llamadas Activas" mode="i"/>
<item key="Call Error" value="Error de Llamada" mode="i"/>
<item key="No Call Key avaiable to make another call" value="No Tecla de llamada Disponible para hacer otra llamada" mode="i"/>
<item key="The requested service is not available at the moment" value="En este momento el Servicio requerido no esta disponible" mode="i"/>
<item key="Conference Error" value="Error de Conferencia" mode="i"/>
<item key="No conference port avaiable to add participant" value="No hay puerto de conferencia para agregar participante" mode="i"/>
<!-- OBiWiFi -->
<item key="OBiWiFi" value="OBiWiFi" mode="i"/>
<item key="Use USB Port 1 for OBiWiFi." value="Use el puerto USB 1 para OBiWiFi." mode="i"/>
<item key="Connected with: " value="Conectado con: " mode="i"/>
<item key="Connecting with: " value="Conectar con: " mode="i"/>
<item key="Failed to authenticate with " value="Fallado para autenticar con " mode="i"/>
<item key="Authentication Failed" value="Falla la Autenticación" mode="i"/>
<item key="Failed to associate with " value="Error al asociar con " mode="i"/>
<item key="Association Failied" value="Asociación no" mode="i"/>
<!-- OBiBT -->
<item key="OBiBT" value="OBiBT" mode="i"/>
<item key="Use USB Port 2 for OBiBT." value="Use el puerto USB 2 para OBiBT." mode="i"/>
<!-- Bluetooth -->
<item key="Yes" value="Sí" mode="i"/>
<item key="No" value="No" mode="i"/>
<item key="Yes For All" value="Sí Para Todos" mode="i"/>
<item key="No For All" value="No Para Todos" mode="i"/>
<item key="Connected" value="Conectado" mode="i"/>
<item key="Paired" value="Emparejado" mode="i"/>
<item key="Connected to " value="Conectado a " mode="i"/>
<item key="Keep plus sign" value="Mantener Signo Más" mode="i"/>
<item key="Trying to connect with " value="Tratando de conectar con " mode="i"/>
<item key="Importing Contacts" value="Importando Agenda de Contactos" mode="i"/>
<item key="The OBi is retrieving contact information from the contected Bluetooth device."
  value="El OBi está obteniendo información de contacto desde el producto Bluetooth conectado." mode="i"/>
<item key="No connected Bluetooth devices!" value="No hay ningún dispositivo Bluetooth conectado!" mode="i"/>
<item key="Contact Access Denied!" value="Acceso a contacto negó!" mode="i"/>
<item key="No Contacts Available!" value="No hay contactos disponibles!" mode="i"/>
<item key="Save new contacts?" value="Guardar nuevos contactos?" mode="i"/>
<item key="Save new contact?" value="Guardar nuevo contacto?" mode="i"/>
<item key="Duplicated" value="Duplicado" mode="i"/>
<item key="Contacts are duplicated!" value="Contactos duplicados!" mode="i"/>
<item key="Contact is duplicated!" value="Contacto duplicado!" mode="i"/>
<item key="Available Devices" value="Dispositivos Disponibles" mode="i"/>
<item key="Enter Bluetooth Passcode:" value="Ingrese el código de acceso Bluetooth:" mode="i"/>
<item key="Telephone number contains a plus sign (+). Keep it?" value="Número de teléfono contiene un signo más (+). Mantenerlo? " mode="i"/>
<!-- Firmware update -->
<item key="Firmware Update" value="Actualizacion del Firmware" mode="i"/>
<item key="Checking for new firmware..." value="Chequeando por Nuevo firmware..." mode="i"/>
<item key="New Firmware available. Press OK to update." value="Nuevo Firmware Disponible. Presione OK para Actualizar." mode="i"/>
<item key="Firmware update not available" value="Actualizacion del Firmware no Disponible" mode="i"/>
<item key="Firmware update in progress..." value="Actualizacion del Firmware en progreso..." mode="i"/>
<!-- Reboot -->
<item key="Reboot" value="Reiniciar" mode="i"/>
<item key="Press OK to reboot the phone." value="Presione OK para Reiniciar el Telefono." mode="i"/>
<!-- Factory reset -->
<item key="Press OK to reset all settings to factory default. Phone will reboot automatically."
   value="Presione OK para Resetear Todos los Ajustes de Fabrica por defecto. El Telefono se Reiniciara Automaticamente." mode="i"/>
<!-- Admin login -->
<item key="Unlock Settings Failed" value="Desbloquear Configuración Error" mode="i"/>
<item key="Invalid Password" value="Contraseña Invalida" mode="i"/>
<!-- OBiTalk -->
<item key="Add Device to OBiTalk" value="Agregue un dispositivo a OBiTalk" mode="i"/>
<item key="Code sent to OBiTalk..." value="Codigo enviado a OBiTalk..." mode="i"/>
<item key="Device not allowed to join!" value="Dispositivo no se le permitió unirse!" mode="i"/>
<!-- Conference bridge -->
<item key="Conf. Bridge Error" value="Error del Puente de Conferencia" mode="i"/>
<item key="Call to Conference Bridge failed" value="Llamada al Puente Conferencia fallado" mode="i"/>
<item key="Conference Bridge ended unexpectedly" value="Puente Conferencia terminó inesperadamente" mode="i"/>
<item key="Last Call Total Charges" value="Gasto Total de la ultima llamada" mode="i"/>
<!-- Network directory -->
```

```
 <item key="Network Directory" value="Directorio Red" mode="i"/>
 <item key="refreshed" value="Actualizado" mode="i"/>
 <item key="Nothing refreshed" value="Nada Actualizado" mode="i"/>
 <item key="Network Directory Search" value="Busqueda de Directorio Red" mode="i"/>
 <item key="No items found for your query" value="No se han encontrado en su búsqueda artículos" mode="i"/>
 <!-- Buddy list -->
 <item key="Buddy List Error" value="Error de Lista de Amigos" mode="i"/>
 <item key="Failed to load buddy list" value="Fracaso en Bajar Lista de Amigos" mode="i"/>
 <!-- Call history -->
 <item key="Clear List" value="Limpiar Lista" mode="i"/>
 <item key="Press OK to remove all entries in the selected call history"
     value="Presione OK para remover todas las entradas en el historial de llamadas seleccionadas" mode="i"/>
 <item key="Duplicate Entry" value="Entrada Duplicada" mode="i"/>
 <item key="Select 'Replace' or 'Add As New' to proceed." value="Seleccione 'Reemplazar' o 'Agregar como Nuevo' para proceder." mode="i"/>
 <!-- Phone book -->
 <item key="Remove Entry" value="Remover Entradas" mode="i"/>
 <item key="Press OK to permanently remove this entry from the Phone Book."
     value="Presione OK para remover permanentemente esta entrada de la Agenda." mode="i"/>
 <item key="Discard Changes?" value="Descartar Cambios?" mode="i"/>
 <item key="Press OK to proceed. Changes to the Phone Book will be discarded."
     value="Presione OK Proceder. Cambios a la Agenda sera descartado." mode="i"/>
 <!-- BT -->
 <item key="Disconnect" value="Desconectar" mode="i"/>
 <item key="Press OK to disconnect" value="Presione OK para desconectar" mode="i"/>
 <item key="Forget" value="Olvidar" mode="i"/>
 <item key="Press OK to forget" value="Presione OK para olvidar" mode="i"/>
 <item key="Connecting" value="Connectando" mode="i"/>
 <!-- Storage explorer -->
 <item key="Remove" value="Remover" mode="i"/>
 <item key="Folders are not removable. Press OK to remove contents within folders." value="Archivos no son removibles pero los contenidos si
seran. Presione OK para continuar." mode="i"/>
 <item key="Folder is not removable. Press OK to remove contents within folder." value="Archivo no es removible pero los contenidos si seran.
Presione OK para continuar." mode="i"/>
 <item key="Removing" value="Removiendo" mode="i"/>
 <item key="Removing folder" value="Carpeta borrado" mode="i"/>
 <item key="Removing file" value="Archivo borrado" mode="i"/>
 <item key="Copying" value="Copiando" mode="i"/>
 <item key="Copying folder" value="Carpeta de copia" mode="i"/>
 <item key="Copying file" value="Archivo de copia" mode="i"/>
 <item key="Invalid characters within" value="Caracteres no válidos dentro" mode="i"/>
 <item key="Failed to create new folder" value="Fracasado en crear nueva carpeta" mode="i"/>
 <item key="Failed to open target file" value="No pudo abrir archivo de destino" mode="i"/>
 <item key="Failed to write target file" value="Dejado de escribir en archivo de destino" mode="i"/>
 <item key="Failed to open source file" value="No pudo abrir el archivo de origen" mode="i"/>
 <item key="Failed to copy selected files!" value="Error al copiar los archivos seleccionados!" mode="i"/>
 <item key="Failed to open source folder" value="No pudo abrir la carpeta de origen" mode="i"/>
 <item key="Failed to remove folder" value="No se pudo quitar la carpeta" mode="i"/>
 <item key="Failed to remove file" value="Fallado en eliminar el archivo" mode="i"/>
 <item key="Failed to remove selected files!" value="No para eliminar los archivos seleccionados!" mode="i"/>
 <item key="Failed to open folder" value="No pudo abrir la carpeta" mode="i"/>
 <!-- Misc -->
 <item key="Progress" value="Progreso" mode="i"/>
 <item key="More" value="Mas" mode="i"/>
 <item key="Disconnected" value="Desconectado" mode="i"/>
 <item key="Secured with" value="Asegurado con" mode="i"/>
 <item key="Not in range" value="Fuera de rango" mode="i"/>
 <item key="Peer to peer" value="De igual a igual" mode="i"/>
 <item key="Ad-hoc" value="Ad-hoc" mode="i"/>
 <item key="Remembered" value="Recordado" mode="i"/>
 <item key="Not Discoverable" value="No detectable" mode="i"/>
 <item key="Available" value="Disponible" mode="i"/>
 <item key="Seconds remaining" value="Segundos Restantes" mode="i"/>
</dictionary>
```

The phone on boot up scans the dictionary directory for all installed dictionary files and prepare a list of language names by extracting the lang attribute in each file. The list of languages are presented to the user under “Preferences/Language” in the phone GUI. On the phone configuration web page the value for the **Language** parameter (under ***User Settings – User Preferences***) is just a string and must be set to one of the available language names.

Phone Book Pictures

These are pictures to associate with each phone book entry. They are stored in the internal folder */people/*. Supported formats are JPEG, PNG, GIF, and BMP. Recommended pixel dimension is no larger than 100Wx100H. Note that the pictures may be shown on the phone screen as part of caller-ID when the peer's number matches an entry in the phone book.

Ring Tones

Ring tone is one of the data types included in the phone customization data package. They are stored in the internal folder */ringtones/*. Ringtone filenames must be ended with the ".wav" extension.

Phone Customization Data Package

Customizable data must be wrapped into one single data package in order to upload onto the phone. The final uploadable data package is a *tar file* that may be optionally gzipped. The size of tar file must be no bigger than 30 MBytes. Inside the tar file, the data should be organized with the same folder structure as the way they are stored in the phone internal storage:

/backgnd	Wall paper (background pictures) and logo.raw (splash screen)
/dict	Dictionary files
/fonts	Text font files
/ringtones	Ring tone files
/people	Phone book contact pictures

As an example, create a directory */home/test/obiphone-data* on a linux machine that constains a folder structure as described above. Copy all the data files to be uploaded to the phone in the corresponding folders in this directory, and issue the following command lines to create the target tar file:

% cd /home/test/obiphone-data

*% tar –zcvf ../obiphone-data.tar.gz **

Note that the tar command must be issued inside of the working directory *obiphone-data*, as OBi Phone will NOT strip the first level in the tar file.

In addition, you can use the following command line to generate the MD5 checksum of the target tar file after it is created:

% md5sum obiphone-data.tar.gz

The MD5 checksum will be used as the version of the target customization data package and it must be configured on the phones also in order to trigger them to download a different version of the package if the currently installed package has a different MD5 checksum from the configured MD5 checksum. This is explained further in the following section.

Uploading Customization Data Package to the Phone

After the tar file of the customization data package is created, it must be placed on a server to be uploaded to the phones when the file is requested from the server by the phones. On the phone side, the downloading of the customization data package from a server is controlled by a number of configuration parameters as described in the table below, not unlike the way phone firmware is updated. As you can tell quickly that these parameters have the same meaning as those similarly named parameters for downloading of firmware.

Note that the phone will attempt downloading of customization data package from the server after firmware update and configuration parameter provisioning are completed. If the MD5 checksum of the downloaded data packages does not match the configured value, the package will be dropped and no data update will be performed. Otherwise, the phone install the downloaded data files into the corresponding folders and then restarts (warm reboot).

Parameter Group	Parameter	Description
Provisioning	**Method**	This parameter controls if and when the phone should execute given **DownloadURL** to download the customization data package (if the installed version is not current). Available choices are: - Disabled = Do not execute **DownloadURL** - System Start = Execute **DownloadURL** just once on system start - Periodically = Execute **DownloadURL** on system start, and then periodically at the interval specified in the Interval parameter Note: First download on system start will be performed after firmware update and configuration provisioning are completed
	Interval	When **Method** is set to `Periodically`, this is the interval in number of seconds between execution of **DowloadURL**. If value is 0, the phone executes **DownloadURL** just once on system start (i.e., equivalent to setting **Method** to `System Start`)
	DownloadURL	In its simplest form, this is the URL to download the Phone Customization Data Package, such as `http://abcd.com/phone/cfg/datapkg.tar`. The full syntax is a script similar to what can be specified in the **FirmwareURL** parameter for firmware update. Refer to the *OBi Device Provisioning Guide* for a description of all the syntaxes of this script.
	MD5Checksum	Standard MD5 checksum of the Customization Data Package (a 32-character string of hexadecimal digits). This value must be provided to the phone and it serves as the version of the target data package. The phone will execute **DownloadURL** only if the currently installed data package has a different MD5 checksum from the value of this parameter.
	Incremental	This Boolean option, if enabled, will cause the phone to install the customization data files without erasing the old data files first. Otherwise, the old data files are erased before saving the downloaded data files.
	DnsLookupType	Choices are: - A Record Only - SRV Record Only - Try Both
	DnsSrvPrefix	Choices are: - No Prefix - With Prefix - Try Both
	Username	(Optional) Username for authentication if **DownloadURL** uses the scheme http:// or https://
	Password	(Optional) Password for authentication if **DownloadURL** uses the scheme http:// or https://

Internal Data Storage Paths for User Preferences Settings

There are a number of configuration parameters to directly or indirectly select one of several internally stored data files for the phone to perform certain tasks, such as background picture or ringtone. This section describes how the built-in and customized data files that support these settings are organized internally so that the administrator can provision these parameter values accordingly.

The data files that are selectable by configuration are organized into three levels: OBi Built-in, ITSP (or Admin) Customized, and User Customized. Each level has it own dedicated storage areas internally. The following table summaries the user settings that utilize these data files and where those data files are stored internally for each level:

Configuration Parameter	Description	Interal Data Storage Folders		
		Phone Built-in	ITSP Customized	User Customized
User Preferences**::** **Language**	The parameter value must match the *lang* attribute of the root element of one of the dictionary xml files found under the folders on the right. When there is a conflict, the user version has the highest priority, then the ITSP version, then the OBi version. On each bootup the folders are scanned to create a list of available languages from the *lang* attributes that users can select from the Preferences menu as their Language.	/data/dict/	/scratch/itsp/dict/	/scratch/phone/dict/
User Preferences**::** **BackgroundPicture**	The parameter value must be the **full internal path name** of a picture file stored in one of the folders on the right or a URL. If a URL is specified in the value, it must start with `http://` or `https://` and the phone will download and cache the data internally until it is power cycled. All available pictures under these folders are also listed under the Preferences menu for users to select as their wall paper picture. The filenames must have one of the following extensions: png, jpg, jpeg, gif, or bmp.	/data/themes/backgnd/	/scratch/itsp/backgnd/	/scratch/phone/backgnd/
User Preferences**::** **DefaultRingtone**	The parameter value must be the **full internal path name** of a wave file stored in one of the folders on the right or a URL. If a URL is specified in the value, it must start with `http://` or `https://` and the phone will download	/data/ringtones/	/scratch/itsp/ringtones/	/scratch/phone/ringtones/

	and cache the data internally until it is power cycled. All available wave files under these folders are listed under a) the Default Ringtone option of the Preferences menu for users to select as their default ringtone, and b) the Ringtone field of the built-in Phone Book appliction for users to choose a ringtone for individual contact in phone book			
There is no configuration parameter that refers to data files in these folders.	All available pictures under these folders on the right are listed under the Picture menu of the built-in Phone Book application for users to select to assign to a contact. The data filenames must on of the following extensions: png, jpg, jpeg, gif, or bmp.	/obi/people/	/scratch/itsp/people/	/scratch/phone/people/
User Preferences**::** **DefaultFont**	The parameter value must be one of the available font family names found in the font.map files in the folders on the right. When there is a conflict, the user version has the highest priority, then the ITSP customized version, then the OBi version.	/data/fonts/	/scratch/itsp/fonts/	/scratch/phone/fonts/

If a parameter value is to specify the filename of an internally stored data file, the full internal path must be specified. For examples:

User Settings – User Preferences**::****BackgrounPicture** = `/scratch/phone/backgnd/CherryBlossom.png`

User Settings – User Preferences**::****DefaultRingtone** = `/data/ringtones/Office A.wav`

OBiPhone XML Applications

With a proprietary XML-formatted scripting language, OBi1000 series IP Phones expose an API to allow third party developers to create apps (i.e. small utility applications) to be downloaded and executed on the phones. We refer to this scripting language as the OBiPhone XML. On the surface the syntax of this language is very similar to those supported by other brands of IP phones. In fact it is more than likely that scripts writtern for these other phones can run as-is on the OBi1000 to produce similar results. At a deeper level, however, OBiPhone XML encompasses rich syntaxes that put a lot of the features available specifically on the OBi1000 phones at the fingertips of the XML App developers.

Although an XML App may be implemented as a single XML script, it typically includes multiple scripts that are executed in sequence to achieve the desired behavior. The execution of an app starts with the first XML script (a.k.a. the "landing" script). Depending on the interaction with the user and other related events happending on the phone, further scripts will be invoked and executed and, eventually, the app exits. The server where the scripts are generated or stored is referred to as the (XML) App Server.

There are two models in which the phone starts the execution of an XML app: a) The pull model where the phone pulls the landing script from a pre-configured URL triggered by some event on the phone (such as user pressing a feature key), and b) The push model where the App Server pushes the script to the phone for execution. The method supported by the phone for pulling is HTTP/HTTPS GET. The method supported by the phone for pushing is SIP Notify with the Event: obixml and Content-Type: application/obixml.

Action URLs – Pull Model

The OBi1000 phones support the pull model by accepting the configuration of the app URL into the phone to associate with a soft key or a feature key, such that when the key is pressed, the corresponding script will be downloaded from the given URL and executed. To be specific, this URL is called an Action URL.

Action URL Feature Key

To invoke an Action URL with a feature key, the feature must be setup with the function `Action URL`, and the URL of the landing script must be specified in the Number field of that feature key. The Name field of the feature may also be configured with a friendly name to be displayed on screen to identify the functionality of the script. For example:

- Function = `Action URL`
- Number = `http://xmlapp-server.obihai.com:8080/xml/testmain.xml?user=jsmith&model=1032`
- Name = `Main Menu`

Action URL Soft Key

To invoke an Action URL with a soft key, include a soft key with the id `acturl` in a soft key set parameter, with the URL specified in a url attribute. For example, the default Home soft key set is: `redial,cfa,dnd,missed|lines`. We can replace the 4th soft key with an action url (all one –line):

```
redial,cfa,dnd,acturl;url="http://10.1.1.123/test.xml?user=abc&amp;model=1032";label="Main Menu"
```

Note that the value of the url attribute MUST be XML-escapled (e.g. & must be specified as &)

SIP Notify – Push Model

As it is typically not feasible for the App Server to send HTTP requests to the phone, OBi1000 only accepts server pushed XML via SIP NOTIFY requests sent to it over one of the enabled SP*n* service channel. The SIP NOTIFY must have the Event: obixml with the Content-Type: application/obixml.

XML App Development

Please refer to the document “OBiPhone XML Application Development Guide” for details on OBiPhone XML syntaxes and and app samples.

Auto Attendant

The OBi1000 has an Auto Attendant (AA) feature which can be invoked by including `aa` as the destination of the inbound call routing rules of a trunk (such as SP1 or OBiTALK) and have incoming calls matching those rules on that trunk routed to the AA. You can specify one of two methods to connect with the AA in the routing rules: a) Ring the AA directly and have it answer the call normally after a configurable delay, or b) Have the AA call back the current caller or another number you designated. To use the AA feature on the phone, the parameter ***Auto Attendant – Auto Attendant 1***::**Enable** must be enabled. OBi1000 supports only 2 simultaneous AA calls. Additional calls routed to the AA will be rejected as busy.

Note that in the phone configuration the AA as described here is referred to as AA1 or Auto Attendant 1. Throughout this document and the phone configuration, AA is the same as AA1. AA2 on the other hand refers to the IVR system that is used for basic phone configuration.

AA Callback Service

The OBi offers two methods for the AA to call you back at the calling number or a number that you picked.

The first method is by statically configuring it in a trunk's **InboundCallRoute**. A rule can be added to the **InboundCallRoute** parameter to have the AA call back the caller's or any other number, if the caller hangs up before the AA answers. The rule should indicate that `aa(`{*callback-number*}`)` is the target destination of the call, where {*callback-number*} is the number that the AA should call back if the caller hangs up before the AA answers the call. For example, the following rule:

```
{(<**1>(14089913313|12121559801)):aa($1)}
```

says that: if 14089913313 or 12121559801 calls, the call is routed to AA. If the caller hangs up before the AA answers, the AA calls the number represented by $1, which is a macro that is expanded into the caller number after processing by the digit map on the left side of the colon. In this particular example, the callback number is the caller's number prepended by **1. The outbound service to be used for the AA to callback is determined according to the ***Auto Attendant – Auto Attendant 1***:: **OutboundCallRoute** parameter.

The parameter ***Auto Attendant – Auto Attendant 1***::**CallbackAnswerDelay** controls the number of milliseconds before AA answers when a callback number is specified as shown in the example. The default value is 10000 ms. Without the ({*callback-number*}) argument, the AA behaves in the normal way and the answer delay is governed by the parameter **AnswerDelay** (in milliseconds).

The second method is by selecting AA option 3 to "Enter a callback number" after the AA answers the call, the caller explicitly enters the number to be called back by the AA. If a valid number is entered, AA says "Thank You" and "Goodbye", and then starts calling back 2s after the current call has ended. If the number entered is invalid, the AA plays the SIT tone followed by an error message. Note that the variable $1 (representing the caller's number) is carried over to the subsequent AA callback call. The AA DigitMap can include $1 to be used in a callback context. For example, the following rule in the AA DigitMap:

(<00:**1$1>|…)

Says that if the AA dials 00, the device will transforms it into the caller's number prepended by **1. In other words, if the caller wants the AA to callback the current number (typically the case), they can simply enter 00# after selecting option 3 on the AA menu. Note that $1 can only be used as part of a substitution element in the digit map; it must not be used for matching elements since its value is unknown.

Automated Attendant:

AA Option	Default AA Announcement	What Happens Next:
1	Press 1 to continue this call.	Ring the phone

2	Press 2 to make a new call.	If "UsePIN" authentication is enabled and the user enters a matching PIN, the OBi Attendant will immediately prompt the user to enter number followed by the pound (#) key. If the entered PIN is not a match, the Attendant will give the user two additional attempts to enter the PIN. If the third attempt does not match, the Attendant will announce a thank you message and disconnect the call.
3	Press 3 to enter a callback number.	If a valid number is entered, AA says "Thank you" and "Goodbye", hangs up, and then callback the number in 2s. It the given number is invalid, AA plays SIT tone followed by an error message. Tips: Caller can simply dial 00# to have the AA call back his current number.

User Recorded Prompts

The OBi supports 10 user recordable prompts that are referred to as the *User1* to *User10* prompt respectively. See the section ***Telephone-IVR-Based Local Configuration*** on how they can be recorded, or the section ***Customized AA Prompts Backup & Restore*** on how they can be duplicated from one device onto another device.

Customizing AA Prompt Lists

The AA does not play individual user prompts directly. Instead it plays a comma-separated list of prompt elements, known as a *Prompt List*. A prompt element can be a user prompt with optional parameters, or a control element. A user prompt is referred as %User<N>% where <N> = 1 – 10. In a prompt list this may be followed by a *;r=<start>-<end>* parameter that specifies the range to play for that prompt, where

<start> = starting time mark in milliseconds; 0 is the default if omitted

<end> = ending time mark in milliseconds; the end of the prompt is the default if omitted

If the r= parameter is omitted, the full range of the prompt is played.

Examples:

%User1%;r=1000 = play User1 prompt starting at 1000ms mark to the end

%User2% = play the entire User2 prompt from start to finish

%User3%;r=1300-3720 = play User3 prompt starting from 1300ms mark to the 3720ms mark

%User4%;r=3200-1200 = does not play anything since <end> is less than <start>

Each prompt list control elements starts with a '&' in a prompt list. The following control elements are supported:

&pause(<duration>) = pause playing for a number of seconds as given by the <duration> parameter

An example of prompt list:

%User1%;r=105,&pause(3),%User5%,%User9%;r=0-1350,&pause(15)

You can replace any of the following AA prompt lists with your own specified prompt lists:

AA Prompt List	System Default	Prompt Be Played
Welcome	Welcome to OBi Attendant	Once, at the beginning when the AA starts
InvalidPin	Invalid PIN	After user enters an invalid PIN
EnterPin	Enter PIN	Prompts user to enter a valid PIN

MenuTitle	Main Menu	Once, after Welcome and before announcing the menu options
Menu	Press 1 to continue this call. Press 2 to make a new all. Press 3 to enter a callback number.	A couple of times after **MenuTitle**
PleaseWait	Please wait while your call is being connected.	Once, after user enters a phone number to call
EnterNumber	Enter number followed by the # key.	Prompts user to enter a valid number after option 2 or option 3 is selected by the user
Bye	Thank you for choosing Obihai Technology. Goodbye.	When user presses * or # key to leave the AA

Voice Gateways and Trunk Groups

Voice Gateway

A gateway in this context is another OBi device that lets incoming OBiTALK callers to call further on one or more of its trunks (such as SP1, SP2, or BT). The caller can call the gateway first with a normal OBiTALK call, get the AA and then dial the target number. For authentication the AA may ask the user to enter a PIN before establishing the second call. This way of dialing is known as 2-stage dialing.

On the other hand, a gateway can be configured on the originating OBi device such that the caller can dial the target number directly without going through the AA. We refer to this method of dialing as direct dialing or 1-stage dialing. Since it is not possible to enter a PIN in the case of direct dialing, a userid/password pair can be configured for the gateway also so that the device can authenticate with the gateway automatically using the HTTP digest method. The HTTP digest authentication is optional. You do not need to provide a user/password if the gateway does not require authentication for direct dialing.

The OBi allows the user to specify up to 8 gateways. Each gateway is addressed using its factory-assigned OBi Number. A gateway is conceptually a trunk with its own DigitMap. You can refer to a gateway and its associated DigitMap with the short trunk name VG*n* and (Mvg*n*) respectively, for *n* = 1, 2, 3, …, 8. VG*n* and (Mvg*n*) can be used in call routing rules and digit maps just like other real trunks.

As an example, you can add the rule `{(1xxx xxx xxxx):vg2}` in ***Phone Settings***::**OutboundCallRoute** to let the device dial out using VGs when the caller dials any 11-digit number starting with 1. On the gateway side, you can add the corresponding rule `{>(1 xxx xxx xxxx):sp1}` in the ***OBiTALK Service***::**InboundCallRoute** to make the call on its SP1 trunk. You can change the last rule to `{(290 333 100|200 444 101)>(1 xxx xxx xxxx):sp1}` if you want to limit the gateway to allow just the two stated caller numbers to make such calls.

A gateway may also be configured with a SIP URL as the access number to be accessed by the device over one of the SP trunks. For example, one can set the gateway access number as `SP1(some-sip-server.mydomain.com)`, or `SP2(192.168.15.111:5062)`, etc. Note that when using a SP trunk to access a (SIP) gateway, the device will:

- Not use the outbound proxy, ICE, or STUN regardless the settings on the SP trunk.
- Use only the device's local address as the SIP Contact, and ignore any natted address discovered by the device.
- Use the gateway's SIP URL to form the FROM header of the outbound INVITE.
- Use the gateway's **AuthUserID** and **AuthPassword** for authentication.
- Apply the symmetric RTP concept.

Trunk Groups

As the name implies, a trunk group is a group of trunks. If a call is routed to a trunk group, OBi picks one of the available trunks from the group to make the call. Availability of trunk is based on:

- Whether the trunk's digit map allows the number to call, AND
- Whether the trunk has capacity to make one more call

Up to 4 trunk groups can be configured on an OBi device. Each trunk group is conceptually another trunk with its own DigitMap. A trunk group and its associated DigitMap are referenced using the short name TG*n* and (Mtg*n*) respectively, where *n* = 1, 2, 3, 4. They can be referenced in other digit maps and call routing rules so that calls may be routed to a particular trunk group.

Only trunks can be added to a trunk group. These include: PP, SP1 – SP6, BT, VG1, VG2, …, VG8, TG1, TG2, … TG4. Note that a TG may include another TG (that is, TG can be recursive). However, you must make sure this does not result in infinite recursion.

LDAP

OBi1000 IP Phones support directory search function with an external server using LDAP. To use this function a LDAP service must be configured on the phone. Users may then invoke the LDAP directory search application by selecting the option from the phone main menu or pressing a soft key.

LDAP Service Setup

The Network Directory option on the main menu of the phone can be pointed to a LDAP (Lightweight Directory Access Protocol) service. The parameters to set up the LDAP service are shown below.

Parameter Group	Parameter	Description
IP Phone – LDAP – Server	**Host**	The hostname can be an IP address or domain name, with optional ldap:// or ldaps:// scheme prefix. For example: `192.168.15.186`, `ldap.forums.com`, `ldap://ldap.testathon.net` are all acceptable hostname formats. If scheme is not specified, ldap:// is implied.
IP Phone – LDAP – Server	**Port**	LDAP Server listening (TCP) port. The starndar port is 389 for `ldap://` and 636 for `ldaps://`. If the port value is 0 or blank, the phone uses the corresponding standard port.
IP Phone – LDAP – Server	**Password**	The password for authentication with the LDAP server, based on the given distinguished name specified in **BindDN**. Note that this parameter is not used for anonymous queries.
IP Phone – LDAP – Search Parameters	**BindDN**	A Distinguished Name (DN) that is authorized to use the LDAP service. If none is specified, the query is regarded as an anonymous one which may or may not be acceptable to the server. The BindDN value is usually derived from a username that typically looks like an email address, such as admin@ldap.example.com. In which case, the corresponding BindDN would be: `CN=admin,OU=users,DC=example,DC=com` Note that in the last example, the DN includes only the last two parts of the domain name for illustration purpose only. The DN may as well include the `DC=ldap` field for example. But the key point is that this value must agree exactly with how it is specified at the server side.
IP Phone – LDAP – Search Parameters	**SearchBase**	This parameter specifies the starting point of the LDAP search. It is a case-insensitive comma-separated list of {object}={value} pairs, where {object} can be any of the followings: • CN Common Name) • OU (Organization Unit) • O (Organization) • C (Country) • DC (Domain) If the value is not specified, the phone by default derives the search starting point from the value of ***LDAP – Server***:: **Host**. For example, if **Host** is `ldap.example.com`, the default SearchBase value is `DC=example,DC=com` Note that it is a common convention to use just the last two parts of the service domain as a search base, but it is not necessary so. The OBi 1000 phones assume this convention when SerachBase is not specified.
IP Phone – LDAP – Search Parameters	**ProtocolVersion**	Protocol version. Either `3` or `2`. `3` is the default.
IP Phone – LDAP – Search Parameters	**TLS_ReqCert**	Control whether to verify the server's certicate on a TLS connection. Choices are: - never - demand Default value is **never**, which means not to verify the server's certificate
IP Phone – LDAP – Search Parameters	**ResultsPerPage**	Default search filter to append to each search. This must be specified as a complete and valid LDAP search filter. For example: **(|(objectclass=contact)(objectclass=person))** Default value is: (objectclass=*)
IP Phone – LDAP – Search Parameters	**DefaultSearchFilter**	Specifies how many resuts to display on screen per page.

		Valid values are: - 20 - 40 - 60 - 80 - 100 Default value is 20
IP Phone – LDAP – Search Parameters	**QueryFields**	A comma separated list of user input LDAP attributes/Display-Name to form the query filter. Each item has three attributes separated by two slashes (/): **{ldap-attr}/{display-name}/{type}** where - **{ldap-attr}** is the standard ldap-attribute-name, such as sn, givenName, telephoneNumber, cn, ...; this is the only required attribute in each field - **{caption}** is optional; this is the caption to display on the screen for the input box. If not specified, the **{ldap-attr}** value is used in its place - **{type}** is either **A** or **N**, for alphanumeric or number type respectively and is case insensitive. If not specified, it is assumed to be **A** For example: **giveName/First Name,TelephoneNumber,Mobile** Default value is: givenName/First Name,sn/Last Name,telephoneNumber/Tel,mobile/Mobile,homePhone/Home
IP Phone – LDAP – Search Parameters	**ResultFields**	Comma separated list of LDAP fields to display for each entry of the search result. Each field has three attributes separated by two slashes (/): **{ldap-attr}/{caption}/{type}** where - **{ldap-attr}** is the standard ldap-attribute-name, such as sn, givenName, telephoneNumber, cn, ...; this is the only required attribute in each field - **{caption}** is optional; this is the caption to display on the screen for the displayed value. If not specified, the **{ldap-attr}** value is used in its place - **{type}** is either **S** , **C**, or **N**, for String, Callable Number, and Picture respectively and is case insensitive. If not specified, it is assumed to be **S** Default value is: cn/Common Name,sn/Last Name,givenName/First Name,telephoneNumber/Tel/y,mobile/Mobile/y,homePhone/Home/y
IP Phone – LDAP – Search Parameters	**NameFieldPreference**	Comma separated list of LDAP attributes to be used as the Caller ID Name to display on screen, ordered by preference. The first non-empty value in the list is used. Default value is: cn,givenName sn
IP Phone – LDAP – Search Parameters	**NumberFieldPreference**	Comma separated list of LDAP attributes to be used as the Number to display on screen and to call by default, ordered by preference. The first non-empty value in the list is ued. Default value is: telephoneNumber,mobile,homePhone
IP Phone – LDAP – Search Parameters	**PhotoFieldPreference**	Comma separated list of LDAP attributes to be used as the photo to display on screen, ordered by preference. The first non-empty value is used. Default value is: thumbnailPhoto
IP Phone – LDAP – Search Parameters	**SortByAttribute**	The LDAP attribute to use for sorting the searh results by the server Default value: cn
SASL Parameters (not applicable for LDAP version 2)		
IP Phone – LDAP – SASL Authentication Parameters	**SASL_AuthMethod**	Method to use for SASL authentication. If the value is Disabled, the phone will only use Simple authentication. Choices are - Disabled - Plain - MD5 Default is Disabled
IP Phone – LDAP – Search Parameters	**SASL_AuthCID**	The authentication ID for SASL authentication. The format of this ID depends on the actual SASL mechanism used.

Client Authentication

LDAP v2 supports `ldap://` and `ldaps://` with Simple Authentication only. LDAP v3 adds support for TLS and SASL Authentication. Simple authentication involves sending the LDAP server the fully qualified DN of the client and the corresponding password in clear-text, which has obivious security issue unless `ldaps://` or TLS is used.

SASL (stands for Simple Authentication and Security Layer) [RFC2222] is a framework for authentication. To use SASL the parameter ***LDAP*** – **SASLAuthMethod** must be set to either **Plain** or **MD5**.

*For more informations on each of these SASL mechanisms, please check for example http://www.openldap.org

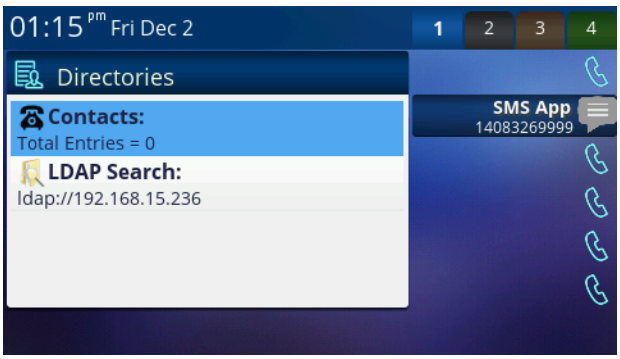
LDAP Directory Search Application

The LDAP application on the phone may be invoked either from the Main Menu or with a soft key (with ID = ldap).

There are two Main Menu option that may be used to invoke LDAP: Directories and Network Directory

Invoke LDAP by Main Menu – Directories Option

Simply include the Main Menu Item ID **directories** in the ***Phone Settings – GUI Menus***::**MainMenu1** parameter then the LDAP optionwill appear under the Directories menu, if LDAP service is properly configured.



Invoke LDAP by Main Menu – Network Directory Option

First you must include the Main Menu Item ID **netdir** in the ***Phone Settings – GUI Menus***::**MainMenu1** parameter. To make the Network Directory option on the Main Menu invoke the LDAP service, use the following settings:

IP Phone Settings – Network Directory::**Enable** = `true` (checked)

IP Phone Settings – Network Directory::**VoiceService** = `LDAP`

Invoke LDP by Soft Key - LDAP

To use a soft key to invoke the LDAP application, add the `ldap` soft key to any of the configurable soft key sets. For example, configure the following value for the **Home** soft key set with the LDAP soft key in the 4th position,

```
redial,cfwd,dnd,ldap
```

Search Fields

By default, the phone presents the following seach fields to the user:

- Last Name (sn)
- First Name (givenName)
- Tel Num (telephoneNumber)

- Mobile Num (mobile)
- Home Num (homePhone)

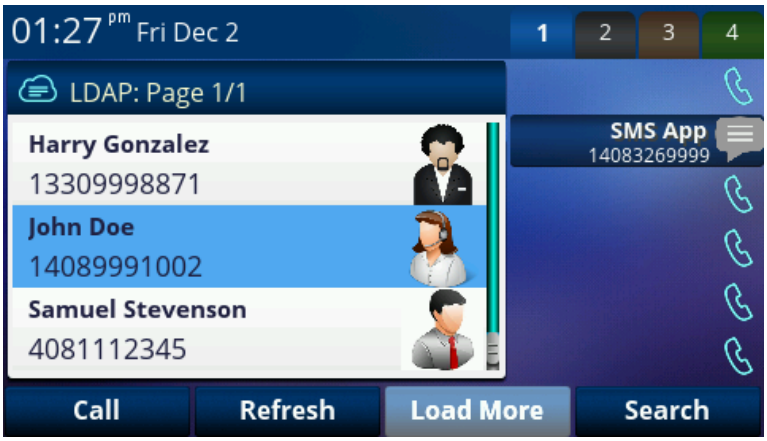
One of more of these search fields may be specified per query; all the search criteria are AND'd together to form the search filter, and the value is case-insensitive. To specify partial match of an attribute value, use a * wildcard character. For example: set Last Name = **s*** to query all entries with the sn that starts with an s or S.

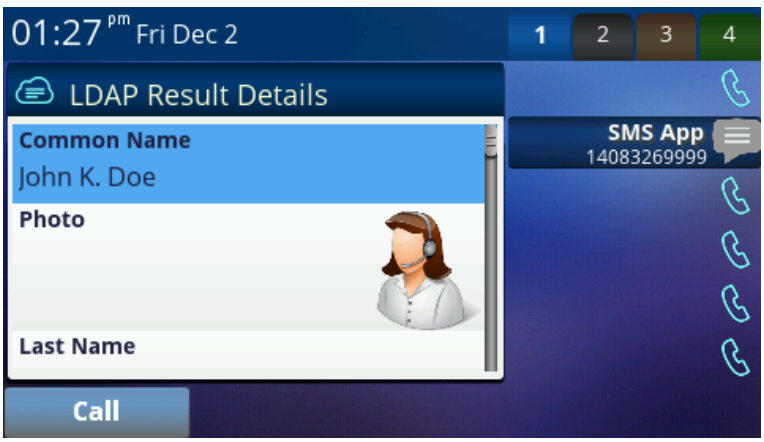
You can customized the search fields to present to the user by specifying the fields in the **QueryFields** parameter. This is a comma separated list of fields. Refer to the LDAP setup parameter table above for the syntax.

Result Fields

The search results are presented in a table where a Name, a Number, and a Picture are shown, when the values are available. Which LDAP attributes in the result are to be used for the Name, Number, and Picture display can be controlled by setting the parameters **NameFieldPreference**, **NumberFieldPreference**, and **PhotoFieldPreference** parameters. Each of these parameters is a comma separated list of LDAP attributes arranged in order of preference. The first attribute in the list that has a non-empty value is used for the corresponding display field.

User can see more attribute values of an entry by highlighting the entry on the top-level search result table and press the OK key on the phone. This brings up the **LDAP Result Details** screen of the selected entry. You can customize the attributes to display on this screen with the parameter **ResultsFields** which is a comma separated list of fields to present to the user. Refer to the LDAP setup parameter table for the syntax of this parameter.





Sorting of Results

The phone relies on the server to sort the results. Sorting is based on a single LDAP attribute which can be customized with the parameter **SortByAttribute**. The default value is **cn**.

Replace the Built-In Phone Book with LDAP

You can replace the built-in phone book with LDAP, such that when the user presses the **phbk** soft key or selects **Contacts** from the Main Menu, it launches the LDAP feature instead of the built-in (local) phone book. Below is the configuration for this:

User Preferences – Phone Book Settings::**ActionURL** = `phone://netdir`

Phone Settings – Network Directory::**VoiceService** = `LDAP`

IP Phone Settings

Settings are divided into the following groups:

- Phone Settings
- Line Keys
- Programmable Keys
- Sidecar 1 and Sidecar 2

Phone Settings

DigitMap and OutboundCallRoute

The **DigitMap** controls what number the user can dial and applies the given transformation to the dialed number. It can refer to the **DigitMap** parameter values in other parameter groups for better readability and organization. The **OutboundCallRoute** determines which service to use based on the dialed number, after validation and transformation by the **DigitMap**.

Primary Line

You can select the Primary Line for the Phone and for the AA, respectively, using the parameters ***Phone Settings***::**PrimaryLine** and ***Auto Attendant***::**PrimaryLine**. The primary line is the default line to use when there is no explicitly selected line and no line-selection prefix (i.e. line access code) has been dialed. For example, when going off-hook to get Dialtone, the phone will try to allocate a call key that is bound to the primary line for the call, if one is available.

You can make one of the SP Services, OBiTALK, OBiBluetooth, or TG1/TG2 as the Primary Line for outbound calls. The Primary Line for the Phone and the Auto Attendant is configured separately. The list below summarizes the choices available for selection as the primary line:

- SP1 Service
- SP2 Service
- SP3 Service
- SP4 Service
- SP5 Service
- SP6 Service
- OBiTALK Service
- OBiBluetooth
- Trunk Group 1
- Trunk Group 2

The user may select a specific Line to use when making a call explicitly by pressing a call key or line monitor key bound to that line or a soft key corresponding to that line. The user may also dial a Line's access code before the destination number. The default service route access codes are defined as:

- ** 1 for SP1
- ** 2 for SP2
- ** 3 for SP3
- ** 4 for SP4

- ** 8 for OBiBluetooth
- ** 9 for OBiTALK

Service route access codes for calling from the Phone can be customized if necessary by modifying ***Phone Settings***::**DigitMap** and ***Phone Settings***::**OutboundCallRoute**. Service route access codes for calling via the Auto Attendant can be customized if necessary by modifying ***Auto Attendant***::**DigitMap** and ***Auto Attendant***::**OutboundCallRoute**.

Note: The phone handles the **PrimaryLine** setting by substituting internally all occurrences of `pli` with the abbreviated name of the trunk named as the primary line in the **DigitMap** and **OutboundCallRoute** parameters of the same parameter group.

Network Directory

The **Enable** option in this group is for the enabling and displaying the Network Directory option on the Main Menu. The **VoiceService** option determines which SP service's network directory function to invoke when the Main Menu option is selected by the user. Note that only one Network Directory option can be shown on the Main Menu. The **Name** parameter is reserved for future use.

Buddy List

The **Enable** option in this group is for the enabling and displaying of the Buddy List option on the Main Menu. The **VoiceService** option determines which SP service's buddy list function to invoke when the Main Menu option is selected by the user. Note that only one Buddy List option can be shown on the Main Menu. The **Name** parameter is reserved for future use.

User Preferences Settings

The **CallForwadUnconditionalFeatureProvider** determines which CallForwardUnconditional parameter the Call Forward option under User Preferences should control; note that this is the same setting the **Call Forward** soft key (in the Home Screen) controls. There is the Phone version and also one version for each voice service. If an SPn service is selected and if the ***SPn – Network Provided Services***::**CallFowardAlways** is `true`, then the user preferences and soft key option reflects and controls the setting of the feature at the server side.

The **DoNotDisturbFeatureProvider** determines which DoNotDisturb parameter the Do Not Disturb option under User Preferences should control. There is the Phone version and also one version for each voice service. If an SPn service is selected and if the ***SPn – Network Provided Services***::**DoNotDisturb** is `true`, then the user preferences option reflects and controls the setting of the feature at the server side.

Page Groups 1 and 2

GroupName is a nickname to refer to the page group; not used anywhere at the moment. **MulticastAddress** and **MulticastPort** define the multicast address of the group to join, and **TTL** sets the TTL value of the outgoing multicast packets. **ParticipantName** is a name to identify to the group the user of this phone via RTCP messages.

To use a page group effectively, there must be a feature key assigned with the corresponding page group function. The user can press to key once to talk to the group, or use PTT to talk if the **PushToTalk** option is also enabled in the page group configuration.

Line Keys

This group is used for the configuration of the 24 (12 on the OBi1032) (Virtual) Line Keys as Feature Keys

Programmable Keys

This group is used for the configuration of the 8 Programmable Keys as Feature Keys

Side Car 1 and Side Car 2

Each group is used for the configuration of the 16 Side Car keys as Feature Keys

Audio Codec Profiles

There are two Codec Profiles available on OBi devices and they are selectable per trunk (OBiTALK, SPn, n = 1 – 6). To select a codec as the preferred codec in this profile, set the priority of that codec to be highest among all the enabled codecs in the profile. Each of the SP and OBiTALK services can be assigned a codec profile in its corresponding configuration. The codec list to use when setting up a call on the underlying service is formed from the list of enabled codecs in the chosen profile and ordered according to the assigned priorities in the profile. For codecs with the same priority setting, the codec appears first on the codec profile web page will be considered as higher priority.

Tone Patterns

Note: Tone and Ring Profile A default settings are set for North American telephone standards. Tone and Ring Profile B default settings are set for Australian telephone standards. Tone profiles for other countries are available for download from the OBiTALK forum.

Tone Profile Features of the OBi Device

The general format for tone profiles follows the following format: [field-1];[field-2];[field-3];...;[field - 6]

Use ";" to separate the configuration fields.

Note that no spaces are allowed to be used in a tone profile pattern.

Field–1 Composition:

This field describes frequency components used for tone synthesis and it supports up to three different frequencies.

The frequency expression is a string of numeric values with the notation '+' or '-'.

The numeric values are the frequency's decimal values in Hz and amplitude in dBm (Maximum 3 dBm).

Different frequencies are separated by ',' .

Example: 350-18,440-18,550+2

The above example illustrates the 1st frequency at 350 Hz with strength at -18 dBm, the 2nd frequency: 440 Hz with strength at -18 dBm and the 3rd frequency: 550 Hz with strength at +2 dBm.

Field–2 Composition:

This field describes the overall tone playback duration in seconds.

The expression is a numeric value, and supports up to 3 decimated digits.

The numeric value can negative, zero, positive, or skipped:

- Negative value: tone plays indefinitely
- Zero value: tone playback is skipped
- Positive value: Normal playback duration
- No value: tone plays indefinitely

Example: 30.234

Meaning: tone playback terminates after 30.234 seconds

Field–3 to Field–6 Composition:

Field - 3/4/5/6 share the same definition, and each field describes one single cadence segment. Together 4 fields form a macro-segment, which will be repeated until tone playback expires.

The expression is a string of numeric values with the special notation '/', '(', ')' and ','.

It has a complete format as below:

t(f_0/on_0+off_0,f_1/on_1+off_1,f_2/on_2+off_2,f_3/on_3+off_3)

t: the cadence segment duration in seconds

- Negative value: tone plays indefinitely
- No value: tone plays indefinitely
- Zero value: the duration of this particular segment is zero
- Positive value: Normal playback duration

f_0/1/2/3: a numerical describe which frequency component(s) are used for the synthesis, and it can be one of following 8 options (0 ~ 7)

- 0: No frequency specified, i.e., silent tone
- 1: The 1st frequency
- 2: The 2nd frequency
- 3: The 1st and 2nd frequencies
- 4: The 3rd frequency
- 5: The 1st and 3rd frequencies
- 6: The 2nd and 3rd frequencies
- 7: The 1st and 2nd frequencies if two or more than two frequency components, or the 1st frequency if only one frequency component is available.

If no value is provided for f_0/1/2/3, it will automatically use the combination of the first one or two available frequency components.

on_0/1/2/3: the tone active time in seconds

- Negative value: Not allowed
- No value: infinite tone active time
- Others: normal tone active time (up to 3 decimated digits)

off_0/1/2/3: the tone inactive time in seconds

- Negative value: Not allowed
- No value: infinite tone inactive time
- Others: normal tone inactive time (up to 3 decimated digits)

Example: 4(1/.3+2.34,3/2+1.5)

The above example illustrates using the first frequency to generate tone for 0.3 seconds, followed by 2.34 seconds of silence, then use a combination of the first and second frequencies to generate tone for 2 seconds, then followed by 1.5 seconds silence. The cadence operates repeatedly for 4 seconds.

Tone Examples:

With these examples, we will show the interpretation of a few common tone patterns:

Dial Tone:

DIAL, "350-18,440-18"

Dial tone is generated as a mixture of two frequency components:

350 Hz at -18 dBm and 440 Hz at -18 dBm

The expiration time is infinite, and tone active time is infinite.

Busy Tone:

BUSY, "480-18,620-18;10;(.5+.5)"

Busy tone is generated as a mixture of two frequency components:

480 Hz at -18 dBm and 620 Hz at -18 dBm

The expiration time is exactly 10 seconds. It has only one cadence segment, which has tone active 0.5 second and tone inactive 0.5 second.

Prompt Tone:

PROMPT, "480-16;10"

Prompt tone is generated from a single frequency component:

480 Hz at -16 dBm. The expiration time is exactly 10 seconds. It has only one cadence segment, which has tone infinite active time.

SIT Tone:

SIT_1, "985-16,1428-16,1777-16;20;(1/.380+0,2/.380+0,4/.380+0,0/0+4)"

Special information tone (SIT) is generated from a set of frequency components:

- 1st frequency: 985 Hz at -16 dBm
- 2nd frequency: 1428 Hz at -16 dBm
- 3rd frequency: 1777 Hz at -16 dBm

The expiration time is exactly 20 seconds. It has only one cadence segment, which includes 4 on-off sections. The segment has infinite repeating time:

- The 1st on-off section: generated by the 1st frequency component, and it has 0.38 tone second active time and 0 inactive time.
- The 2nd on-off section: generated by the 2nd frequency component, and it has 0.38 tone second active time and 0 inactive time.
- The 3rd on-off section: generated by the 3rd frequency component, and it has 0.38 tone second active time and 0 inactive time.
- The 4th on-off section: only generate silence since no frequency component is specified. It has tone 0 second active time and 4 seconds inactive time.

Stutter Tone:

STUTTER, "350-18,440-18;10;.6(.1+.1);(/)"

Stutter dial tone is generated from a mixture of two frequency components:

350 Hz at -18 dBm and 440 Hz at -18 dBm. The expiration time is exact

10 seconds. It has two cadence segments.

- The first segment: includes only one on-off sections, on 0.1 second and off 0.1 second, and on-off repeats for 0.6 second.
- The second segment: include one on-off section, and has infinite repeating time and infinite tone active time.

Ringtones and Ring Patterns

The general format of an OBi Ring Profile is as follows: [field-1];[field-2];...;[field - 5]

Use the ";" to separate up to five (5) configuration fields.

Please note that no spaces are allowed to be used in a tone profile pattern.

Field–1 Composition:

Field-1 describes the overall ringing duration in seconds.

The expression is a numeric value, and supports up to 3 decimated digits.

The numeric value can negative, zero, and positive:

- Negative value: Ringing lasts indefinitely
- No value: Ringing lasts infinitely
- Zero value: Ringing is skipped
- Positive value: Normal ringing duration

Example: 30.5

The above example illustrates a ringing tone that terminates after 30.5 seconds.

Field –2 to Field –5 Composition:

Field - 2/3/4/5 share the same definition, and each field describes one single cadence segment. Together, the four (4) fields form a macro-segment, which will be repeated until ringing expires.

The expression is a string of numeric values with the special notation '**(**' , '**)**' and '**,**'

It has the format as per the following construct: t(on_0+off_0,on_1+off_1,on_2+off_2,on_3+off_3)

t: The cadence segment duration in seconds.

- Negative value: Ringing indefinitely
- No value: Ringing indefinitely
- Zero value: Ringing is skipped
- Positive value: Normal ringing duration

on_0/1/2/3: The ring active time in seconds.

- Negative value: Not allowed
- No value: Infinite ring active time
- Others: Normal ring active time (up to 3 decimated digits)

off_0/1/2/3: The ring inactive time in seconds

- Negative value: Not allowed
- No value: Infinite ring inactive time
- Others: Normal ring inactive time (up to 3 decimated digits)

Example: 4(.3+2.34,2+1.5)

The above example illustrates a ringing tone comprised of two segments. Ringing is active for 0.3 seconds, followed by 2.34 seconds of silence, then ringing for 2 seconds, and followed by 1.5 seconds of silence.

The above cadence operates repeatedly for 4 seconds.

Star Code Profiles

Star codes are short sequences of digits where each sequence serves as a command to the OBi Device to perform certain operation. Each sequence usually starts with the * key followed by a 2-digit code (such as *69), hence the term star code. A typical operation to carry out is to set the value of one or more configuration parameters. At present the OBi device allows user to issue star code from the PHONE port only; user issues a star code the same way he dials a number to make a call. In OBi every star code and its operation are defined with a short *Star Code Script* parameter. The set of star codes that can be dialed from the PHONE port is collectively referred to as a Star Code Profile.

The OBi has two star code profiles available in its configuration, known as Start Code Profile A and B respectively. Each profile has 30 star code script parameters, known as Code1 to Code30. You can select which star code profile to use by setting ***Phone Settings***::**StarCodeProfile** to *A* or *B*, or *None* if star code is not to be used.

A star code script is defined with the help of a number of predefined variables and actions. Each variable represents one or one group of configuration parameters. An action can be checking or setting the value of a variable, collecting a phone number from the user, or calling a certain number.

Star Code Script Variables (*VAR*)

A star code script variable or *VAR* can be trunk specific or global (non-trunk specific). The general format of a global variable is $var. The general format of a trunk specific variable is *TK*($var) , where *TK* is the abbreviated name of a trunk (SP1-SP6, BT, or PP). If *TK* is not specified for a trunk-specific variable, it implies all the applicable trunks in the system.

Note that SP*n* is the SP*n* Service where *n* = 1-6, BT the OBiBluetooth Service, and PP the OBiTALK Service. Each service is also referred to as a “trunk” in this document.

Here is a list of the supported $var:

$CFA = call forward unconditional enable (trunk specific; admissible value: 0 for disable, 1 for enable)

$CFB = call forward busy enable (trunk specific; admissible value: 0 for disable, 1 for enable)

$CFN = call forward no-answer enable (trunk specific; admissible value: 0 for disable, 1 for enable)

$CFAN = call forward unconditional number (trunk specific; admissible value: a token representing a call forward number)

$CFBN = call forward busy number (trunk specific; admissible value: a token representing a call forward number)

$CFNN = call forward no-answer number (trunk specific; admissible value: a token representing a call forward number)

$MWS = message waiting state (trunk specific; admissible value: 0 for no new messages, 1 for one or more new messages)

$DND = do-not-disturb enable (trunk specific; admissible value: 0 for disable, 1 for enable)

$BAC = block-anonymous caller enable (trunk specific; admissible value: 0 for disable, 1 for enable)

$BCI = block outbound caller-ID enable (trunk specific; admissible value: 0 for disable, 1 for enable)

$CWA = call-waiting enable (global; admissible value: 0 for disable, 1 for enable)

$BCI1 = block caller-ID once (global; admissible value: 1 for enable)

$UBCI1 = unblock caller-ID once (global; admissible value: 1 for enable)

$LBM1 = Loopback media (audio samples) once in the next call

$LBP1 = Loopback RTP packets once in the next call

$CDM1 = Codecs to enable in the next call (temporarily overriding any codec preferences in device configuration). Each bit of its value represents one audio codec:

- Bit0 (LSB) = G711u
- Bit1 = G711a
- Bit2 = G726r16
- Bit3 = G726r24
- Bit4 = G726r32
- Bit5 = G726r40
- Bit6 = G729

$LDN = last dialed number (for redial) (global; read only)

$BAR1 = Enable Barge-In 1 on the next call (global; admissible value: 1 for enable)

$Bxrn = Blind Transfer Target Number (global; admissible value: a token representing the target number)

$LCR = last caller's number (for call return) (global; read only)

$SPD[*n*] = number for the speed dial *n* (*n* = 1 – 99) (global; admissible value: literal or token representing a phone number)

$CODE = the digit(s) representing the variable part of a star code (see examples below; read only)

Variable names are CASE INSENSITIVE.

Star Code Script Actions (*ACT*)

The general format of an action: *ACT*(*par*, *par*, ….)

The following actions are supported:

- set(*VAR*,*token*) = Set the given *VAR* to the value represented by *token*.
- call(*token*) = Call the number represented by *token.*
  - ***Phone Settings***::**OutboundCallRoute** will be applied when making the call (but not the **DigitMap**)
- rpdi(*token*) = repeat dial the number represented by *token*
- coll(*VAR*) = collect a number from the user and store it as the value of the parameter(s) represented by *VAR*.
  - The number is collected with ***Phone Settings***::**DigitMap** applied
- say(*token*) = display the value represented by *token* in an on-screen notification message
  - Values are announced as a list of alphabets or numbers

where *token* can be a literal (such as 1234) or another variable (such as $CFAN or SP1($CFBN))

- btdscvr(*n*) = Enable OBiBT discovery for 2 minutes
  - *n* = 0 or 1 for OBiBLueTooth 1 and OBiBlueTooth 2 respectivvely
- wifiap = Enable WiFi AP mode for quick WiFi configuration from a web browser
- blst = Add the number of the last caller to the X_BlockCallers list

Action names are CASE INSENSITIVE.

Star Code Script Format

General Format: code, name, action1, action2, action3, …

- code = the star code, such as *72. It may contain a variable part enclosed in parenthesis, such as *74(x|xx)
  - The variable part as entered by the user are stored in the variable $CODE
- name = a descriptive name of the function of this star code, such as *Call Forward Unconditional*
- action1, action2, … = a valid action with parameters

Actions are carried out one-by-one in the order as specified in the script.

Restrictions:

- At most 1 *coll* action per code.
- Either 1 *say* or 1 *call* action at most per code, and it must be the last action in the script.

Star Code Script Examples

The following examples are taken from some of the default star code scripts in the OBi device.

*69, Call Return, call($LCR)

- Calls the number of the caller who rings the PHONE port last time

*07, Redial, call($Ldn)

- Redials the last dialed number

*72, Call Forward Unconditional, coll($cfan),set($cfa,1)

- Collects a number from the user according to the DigitMap. Then set the CallForwardUnconditionalNumber on all trunks to the collected value, and set the CallForwardUnconditionalEnable on all trunks to Yes
- To modify the script to enable CallForwardUnconditional on SP1 only, change it to

*72, Call Forward Unconditional SP1, coll(SP1($cfan)),set(SP1($cfa),1)

*67, Block Caller ID Once, set($BCI1,1)

- Enable masking of caller ID information once for the next call on any trunk

*74(x|xx), Set Speed Dial, coll($Spd[$code])

- After user dials *74, OBi expects one or two more digits from the user which represent a speed dial slot index (1 to 99). The 1 or 2-digit variable part is stored in the variable $code.
- OBi device then plays a prompt tone and proceeds to collect a number from the user according to the DigitMap. Finally OBi stores the collected number in the given speed dial slot. If the slot already has a number specified, it will be overwritten quietly with the new value.

*75(x|xx), Check Speed Dial, say($Spd[$code])

- After user dials *75, OBi expects one or two more digits from the user which represent a speed dial slot index (1 to 99). The 1 or 2-digit variable part is stored in the variable $code.
- The phone displays a message box on the screen to show the number stores in the speed dial slot.

Default Star Codes

The OBi device supports service features via the handset connected to the PHONE port. The following Star Codes can be used to access the indicated features. OBi Star Code Enabled Features Apply to All Voice Services.

*03, Request peer device to loopback media in the next outbound call
*04, Request peer device to loopback RTP packets in the next outbound call
*05, Tell device to periodically redial the last called number until the called party rings or answers

*06, Cancel the last repeat dial request
∗07 Redial
∗69 Call Return
∗81 Block Caller ID (Persistent Mode)
∗82 Unblock Caller ID (Persistent Mode)
∗67 Block Caller ID (One Time)
∗68 Unblock Caller ID (One Time)
∗72 Call Forward All (Enter Number + #)
∗73 Disable Call Forward All
∗60 Call Forward on Busy (Enter Number + #)
∗61 Disable Call Forward in Busy
∗62 Call Forward on No Answer (Enter Number + #)
∗63 Disable Call Forward No Answer
∗77 Block Anonymous Calls
∗87 Unblock Anonymous Calls
∗56 Enable Call Waiting
∗57 Disable Call Waiting
∗78 Do Not Disturb – Turn On
∗79 Do Not Disturb – Disable
∗74 Speed Dial Set-Up (Enter SD No. [1-99] then Tel No. + #) ∞
∗75 Speed Dial Read-Back (Enter SD No.)
*76, Clear A Speed Dial
*96, Barge In (i.e. request called phone to auto-answer)
*27, Tunr easy WIFi Setup Mode (i.e. Phone acts temporarily as AP to allow WiFi configuration)
*4711, Use G711 Only on the next outbound call
*4729, Use G729 Only on the next outbound call
*28, Make OBiBluetooth discoverable for 2 minutes

∞ Note: Be careful with the Speed Dial Set-Up as this will conflict with the Speed Dials set-up on the OBiTALK portal. The Speed Dials that are set-up on the OBiTALK portal will always overwrite anything set-up via the phone connected to the OBi.

User Settings

Speed Dial Numbers

Each OBi device supports 99 speed dial numbers. The 99 speed dial slots are numbered from 1 to 99 and are invoked by dialing a 1 or 2-digit number corresponding to the slot number. Speed dials may be dialed from the phone or via the Auto Attendant. Note that the 2-digit numbers “01”, “02”, …, “09” are not admissible; you must dial the 1-digit number “1”, “2”, …, “9” for slot number 1-9.

Speed dial value can be set using the configuration web page, remote provisioning, or star code (see the *Star Code Section* in this document for more details). The value may be a number just like the one you normally dial, with or without any service access code prefix, such as: **9200112233, **214089991123, 4280913, etc. It may also include explicit trunk information with the general format TK(number), where TK= SP*n* (*n*=1 – 6), BT, or PP. For example, PP(ob200112233), SP2(14089991123), BT2(4280913), etc.

If trunk information is *not* specified in the speed dial entry, OBi device applies **DigitMap** and **OutboundCallRoute** when making the call. Otherwise neither **DigitMap** nor **OutboundCallRoute** is applied.

Using Speed Dial Number as Ad Hoc Gateway

If an external gateway does not require authentication, its access number can be stored in one of the 99 speed dial slots to allow ad hoc direct dialed gateway calls. To do this, the user dials the gateway’s speed dial, followed by a *, followed by the target number. That is <gateway-speeddial> * <target-number>. For example, the gateway access number pp(ob200333456) is stored at speed dial 8, and the user can dial 8*14085551234 to call 14085551234 using the given gateway.

Note: At the present time, only gateways that are accessed with an OBi number can be used this way.

Call Routing

Trunks, Endpoints, and Terminals

An OBi1000 Phone is also a Voice Service Bridge (VSB) that supports multiple voice services. It can bridge calls across any of the supported services. By a call bridge we refer to a voice connection connecting two calls on the same or different voice services. The OBi1000 allows 4 concurrent independent call bridges. The following matrix shows the possible call bridge connections.

Supported 2-way call bridges on the OBi1000

	SP1 Service	SP2 Service	SP3 Service	SP4 Service	SP5 Service	SP6 Service	OBiTALK Service	OBiBlue tooth
SP1 Service	yes	yes	yes	yes	yes	yes	yes	yes
SP2 Service	yes	yes	yes	yes	yes	yes	yes	yes
SP3 Service	yes	yes	yes	yes	yes	yes	yes	yes
SP4 Service	yes	yes	yes	yes	yes	yes	yes	yes
SP5 Service	yes	yes	yes	yes	yes	yes	yes	yes
SP6 Service	yes	yes	yes	yes	yes	yes	yes	yes
OBiTALK Service	yes	yes	yes	yes	yes	yes	yes	yes
OBiBluetooth	yes	yes	yes	yes	yes	yes	yes	no

Each supported service is also referred to as a *trunk* (a traditional telco term for a physical wire or wires that deliver phone services to homes or businesses). Each trunk is represented with 2-letter abbreviation and a 1-based instance identifier:

- SP1 = the SP1 Voice Service (with ITSP A, B, C, D, E or F)
- SP2 = the SP2 Voice Service (with ITSP A, B, C, D, E or F)
- SP3 = the SP3 Voice Service (with ITSP A, B, C, D, E or F)
- SP4 = the SP4 Voice Service (with ITSP A, B, C, D, E or F)
- SP5 = the SP5 Voice Service (with ITSP A, B, C, D, E or F)
- SP6 = the SP6 Voice Service (with ITSP A, B, C, D, E or F)
- PP1 = the OBiTALK Service
- BT1 = the OBiBluetooth Service (with built-in Bluetooth or via an OBiBT dongle connected to USB Port 2)

The instance identifier may be omitted if it is equal to 1; hence BT is equivalent BT1, PP is equivalent to PP1, etc. These short-hand notations are used heavily in configuring the OBi device, as found in call routes, call forward numbers, and speed dials parameters. Unless stated otherwise, the abbreviated trunk names are case insensitive.

The Phone (PH) and the AA, are the entities in an OBi1000 phone that calls can terminate (i.e., starts or ends there), as opposed to the trunks, which rely on the corresponding service providers to terminate the call. In this document we refer to the Phone and AA as *endpoints*. Like the trunks, each endpoint is represented by a 2-letter abbreviation and a 1-based instance identifier:

- PH = The Phone
- AA = The Auto Attendant

Unless stated otherwise, abbreviated endpoint names are case insensitive. A trunk or an endpoint is also referred to as a *Terminal* in this document.

The following matrix shows the possible call connections between the endpoints and the trunks:

Supported endpoint calls on the OBi:

	Any Trunk	**PH1**	**AA**
Any Trunk	n/a	yes	yes
PH	yes	no	yes
AA	yes	yes	no

Call Routing – The OBi Way

Call Routing is the process by which the OBi Device sets up a call bridge or a (endpoint) call based on such information as: the trunk on which the call originates, the caller's number, the called number and so on. Call Routing Rules are parameters used to instruct the OBi device how to route calls. A call may transform into a call bridge or an endpoint call after being routed by the OBi according to the given routing rules.

Every call has to originate from somewhere. From the device's perspective, calls originating from the trunk side are considered Inbound Calls, while calls originating from an endpoint are Outbound Calls. The call routing rule syntaxes for inbound calls and outbound calls are slightly different and we shall explain them separately below. Call Routing Rule configuration relies heavily on digit maps. If you are not familiar with how digit maps work yet, please read the *Digit Map Configuration Section* in this document first. You can also read the ***OBi-DigitMap Call Route Tutorial*** document that is available for download from www.obihai.com/docs-downloads

Inbound Call Route Configuration

Every trunk has a corresponding **InboundCallRoute** in the OBi device configuration. It is a comma separated list of rules where each rule is also surrounded by a pair of curly brackets { }. No extra white spaces are allowed. These rules tell the OBi how to handle an inbound call, such as sending it to the Phone (and ringing the attached phone(s)), sending it to the Auto Attendant for further routing (interactively with the caller), or making another call on a specific trunk to bridge with this call.

The general format is:

InboundCallRoute := *rule* **OR** {*rule*},{*rule*},….

Note that the curly braces may be omitted if there is only one rule in the route. The **OR** operator is NOT part of the parameter syntax; it is used here to separate alternative values only.

A rule has the following format:

Inbound Call Route Structure						
{rule},{rule},{rule},{rule}						
rule =						
{peer-list			:	terminal-list}		
peer-list =						
{(peering), (peering), (peering)			:	(terminal), (terminal)}		
peering =				terminal =		
{(caller-list	>	callee-list)	:	PHx OR AAx OR LIx(arg) OR SPx(arg) OR PPx(arg)}		
caller list =		callee-list =		arg=		
(caller|caller|caller	>	callee|callee)		cid	>	target
caller =		callee =		cid =		target =
number OR digit-map OR ? OR @ *(?=anonymous, @=any #)*		number OR digit-map OR @		digit-map or spoofed-caller-number *(the outbound CallerID)*	>	digit-map or number-to-call

Notes:

- *Terminal-list* can be empty, which means to block this call. The preceding ':' cannot be omitted. Up to 4 *terminals* may be specified in the list. The listed *terminals* will be called/rung by OBi simultaneously; we refer to this operation as *forking* the call. A terminal may be a trunk or an endpoint.
- Abbreviated terminal names are case-insensitive
- *number and number-to-call* are literal strings, such as 14089991234
- *digit-map* is just any proper digit map, such as (1xxx|xx.); make sure to include the enclosing parentheses
- *spoofed-caller-number* is a literal string, such as 14081112233, to be used as the caller number for making a new call on the specified trunk
- (M*label*) is a named digit map, where *label* is the abbreviated name of any terminal that has a digit map defined: SP1–SP6, BT, PP, PH, or AA
- $1 is an internal variable containing the value of the caller number of this inbound call, after any digit map transformation in the matched *caller* object of the matched *peering* object in the *peering-list*.
- $2 is an internal variable containing the called number of this inbound call, after any digit map transformation in the matched *callee* object of the matched *peering* object in the *peering-list*.

More notes on *peering-list* and *peering* objects:

- *Peering-list* is optional in **InboundCallRoute**. If *peering-list* is empty, the succeeding ':' can be omitted also. An empty *peering-list* implies a single *peering* object whose *caller* object list matches any caller number. That is, the **InboundCallRoute** values listed below are all equivalent
  - ph
  - {ph}
  - {:ph}
  - {?|@>@:ph}
- *Callee-list* in a *peering* object can be empty. It implies the *callee* object @, meaning any called number. The preceding '>' can be omitted if callee-list is empty.
- *Caller-list* in a *peering* object can be empty. It implies the *caller-list* @|?, meaning any caller number including anonymous. The succeeding '>' cannot be omitted if caller-list is empty but not the callee-list

More notes on the *arg*, *cid*, and *target* objects:

- The *cid* object inside an *arg* object is optional. If omitted, it implies no caller-ID spoofing when making the call on the specified trunk. The succeeding '>' can be omitted is *cid* is omitted
- The *target* object inside an *arg* object is optional. If omitted, it implies the *target* $2, which means to call the original called number after applying any necessary digit map transformation implied by the rule. The preceding '>' cannot be omitted if *target* is omitted but cid is not
- *arg* object is optional. If omitted, it implies the *arg* with the *target* $2 and no *cid*. If *arg* is omitted, the succeeding parentheses ( ) can be omitted also.

An inbound call matches a rule if its caller-number/callee-number matches one of the *peering* objects of the rule. *Peering* objects are tested in the order left and right, and the first matched *peering* object will win. Rules are also checked in the order left to right, and the first matched rule will win. Therefore it is important that you place the more specific rules first in the **InboundCallRoute** if multiple rules can potentially match the same inbound call.

InboundCallRoute Examples:

1) `ph` **OR** `{ph}` **OR** `{:ph}` **OR** `{@|?>@:ph}` (all equivalent)

It says: Ring the phone (only) for all incoming calls. This is the default **InboundCallRoute** for all trunks.

2) `{14081223330|15103313456:ph,aa},{(1800xx.|1888xx.)|?:},{ph}`

It says: Ring the phone and AA for calls coming from 1 408 122 3330 or 1 510 331 3456, block all 800, 888, and anonymous calls, and ring just the phone for all other calls

3) `{(x.4081113333|x.4152224444):aa},{ph}`

It says: Ring the AA for calls coming from any number that ends with 408 111 3333 or 415 222 4444, and ring the phone for all other calls. Be sure to include the enclosing parentheses in this example since "x." is a digit map specific syntax.

4) `{200123456:aa},{sp1(14083335678)}`

It says: Ring the AA for calls coming from 200123456. For all any other call, bridge it by calling 1 408 333 5678 using SP1 Service

Outbound Call Route Configuration

Every endpoint has an **OutboundCallRoute** parameter in the OBi device configuration. It tells the device where to send the call when the endpoint attempts to make a call. Endpoints may call each other or an outside number using one of the trunks. The **OutboundCallRoute** syntaxes are almost identical to those of the **InboundCallRoute**; the differences are mainly in the implied value when an optional field is omitted, no *caller* objects and one and only one terminal object per terminal-list in an **OutboundCallRoute**. Forking is not supported when routing outbound calls.

The general format is:

OutboundCallRoute := *rule* **OR** {*rule*},{*rule*},….

Note that the curly braces may be omitted if there is only one rule in the route. The **OR** operator is NOT part of the parameter syntax; it is used here to separate alternative values only.

A rule has the following format:

Outbound Call Route Structure		
{rule},{rule},{rule},{rule}		
rule =		
{callee-list	:	terminal}
callee-list =		terminal =
{callee|callee	:	PHx, AAx, SPx, LIx, or PPx}
callee =		
number OR (digit-map) OR @	*(@=any number)*	

Notes:

- A terminal may be a trunk or another endpoint.
- Abbreviated terminal names are case-insensitive
- *number and number-to-call* are literal strings, such as 14089991234
- *digit-map* is just any proper digit map, such as (1xxx|xx.); make sure to include the enclosing parentheses
- *spoofed-caller-number* is a literal string, such as 14081112233, to be used as the caller number for making a new call on the specified trunk
- (M*label*) is a named digit map where *label* is the abbreviated name of any terminal that has a digit map defined: SP1–SP6, BT, PP, PH, or AA
- $2 is an internal variable containing the called number of this outbound call, after any digit map transformation in the matched *callee* object
- *Callee-list* can be empty, which implies the single *callee* object @, which means any called number. The succeeding ':' can be omitted also when *callee-list* is empty

More notes on the *arg*, *cid*, and *target* objects:

- The *cid* object inside an *arg* object is optional. If omitted, it implies no caller-ID spoofing when making the call on the specified trunk. The succeeding '>' can be omitted if *cid* is omitted.
- The *target* object inside an *arg* object is optional. If omitted, it implies the *target* $2, which means to call the original called number after applying any necessary digit map transformation implied by the rule. The preceding '>' cannot be omitted if *target* is omitted but not the cid.
- *arg* object is optional. If omitted, it implies the *arg* with the *target* $2 and no *cid*

An outbound call matches a rule if its called number matches one of the *callee* objects of the rule. *Callee* objects are tested in the order left and right, and the first matched *callee* will win. Rules are also checked in the order left to right, and the first matched rule will win. Therefore it is important that you place the more specific rules first in the **OutboundCallRoute** if multiple rules can potentially match the same outbound call.

Note that every endpoint also has a digit map defined. The user dialed number is completely processed with the endpoint's digit map first before it is passed to the **OutboundCallRoute** for routing decision. Therefore the number used for matching call routing rules has already incurred the transformations, if any, implied by the digit map. Remember this fact when crafting your own **OutboundCallRoute**.

OutboundCallRoute Examples:

1) `sp1` **OR** `{SP1}` **OR** `{:SP1}` **OR** `{@:Sp1}` (all equivalent)
This rule says: Make all calls using SP1 Service, without any caller-id spoofing or digit transformation

2) `{**0:aa},{***:aa2},{(Mpli):pli},{(<**1:>(Msp1)):sp1},{(<**2:>(Msp2)):sp2},{(<**8:>(Mbt)):bt},{(<**9:>(Mpp)):pp}`

This is the default ***Phone Settings***::**OutboundCallRoute**. It says:

- Dial **0 to invoke AA
- Dial *** to invoke the local device configuration IVR (a.k.a AA2)
- (Mpli) and pli will be substituted with the abbreviated name of the **PrimaryLine** value
- Use SP1 Service to call all numbers that start with **1 and subsequent digits matching SP1 Service's **DigitMap**. Remove the **1 prefix from the resulting number before making the call
- Use SP2 Service to call all numbers that start with **2 and subsequent digits matching SP2 Service's **DigitMap**. Remove the **2 prefix from the resulting number before making the call
- Use the LINE port to call all numbers that start with **8 and subsequent digits matching OBiBluetooth Service's **DigitMap**. Remove the **8 prefix from the resulting number before making the call
- Use the OBiTALK Service to call all numbers that start with **9 and subsequent digits matching OBiTALK Service's **DigitMap**. Remove the **9 prefix from the resulting number before making the call

Digit Map Configuration

A digit map can be used to match digits to ensure a complete number is dialed, transform dialed digits and block numbers from being dialed. It is structured as a series of rules that are read from left to right. The OBi will apply the first rule that matches the format of the dialed number, so it's important you get your rule order correct. Each digit map is composed of one or more rules surrounded by round brackets ( ) these brackets MUST NOT be omitted.

We can define a digit map as a group of rules written in the following fashion:

Digit Map = (rule | rule | rule | rule | rule)

We use a vertical bar | to separate each rule. Once again, the digit map must include enclosing brackets ( ) or the device will not read the entered text as a digit map.

You can include "white space" within your digit map rule to make it more readable - the OBi will ignore the spaces when reading the rules in the digit map. This can help make it easier to read your rules. The following example shows a rule to match phone numbers dialed to the London area code from within the United Kingdom:

(020[378]xxxxxxx) *is equivalent to* (020 [378]xxx xxxx)

We will cover the syntax used in this example further on.

Digit Map Elements

The digit map rule serves to match numbers based on the characteristics of the form and content the entered number (or alphanumeric string, for example, when entering a SIP URI on the OBi IP Phone).

A rule is made up of a series of elements. Each element of the rule is matched from left to right in sequence with the entered string of numbers (or characters).

The following series of tables detail all available elements with which to create a rule:

Any combination of: 0-9 * # + - A-Z a-z *Excluding:* m M s S x X	**Literals** It matches digit sequences with exactly the same literals. m, M, s, S, x, X which have special meaning in the digit map syntax. Use literals to explicitly match a string.
Example: To explicitly match the New York phone number 1-212-555-7722 as dialed from within North America (ie according to the NANP) we would create the following rule from literals: **12125557722** *or* **1 212 555 7722** *as spaces may be added for clarity*	

' '	**'Quoted Literals'** Everything inside a pair of single quotes is treated as a literal except for the single quote ' character itself.
Example: To explicitly match the SIP URI matt@sipservice.com we need to include ' ' as the address includes reserved characters. We would write our matching rule as: **'matt@sipservice.com'**	

x	x Wildcard The "lowercase x" – x – wildcard matches any digit from 0-9 It is important to note that x is CASE SENSITIVE.
Example: If we consider phone numbers within a specific area code, all of the same length, we can write a rule	

that explicitly matches digits at the start of the number followed by any sequence of digits of a defined, fixed length. When dialing a number in the Dalton, Georgia area code of 706 from within North America, we would dial 1 706 then the following 7 digits of the number, as such we would write our matching rule as follows:
1 706 xxx xxxx

.	. Matching Function The "full stop" or "dot" - . - matching function matches 0 or more x, X or @ It's typically used to capture a string of entered numbers or characters of arbitrary length.

Example:
The expression xx. would catch any number that is entered with 1 or more digits. Although we have used two x's in our rule, the dot implies 0 or more x, so in the event of 0*x the rule xx. caters for a single digit number being entered. Let's say we want to match any international numbers dialed to New Zealand (country code 64) but don't want to define all the possibilities of the NZ number plan within our digit map; in this case we can write one of the following rules:
011 64 xx. *Using US international dialing notation*
0011 64 xx. *Using Australian international dialing notation*
00 64 xx. *Using Standardized international dialing notation*

[ ]	Set "Square brackets" – [ ] – enclose a set of numerals and/or characters that are used to <u>match a single digit or character</u>. It's useful to use a set to match specific digits that form part of a number. Alphanumeric and wildcard characters are allowed inside a set, such as [x], [X#], [@#], [a-zA-Zx]

Example:
Let's look at specifically matching the numbers 1,2,5,6,7 and 8. We can write this set as [125-8] where we have specified the digits 1 and 2, then written the numbers 5,6,7,8 as the sequence 5-8. To put this into context, let's look at the London number range where each number can take the form of 020 3xxx xxxx, 020 7xxx xxxx or 020 8xxx xxxx – while we could write each of these as individual rules, it is tidier to represent them as follows:
020 [378]xxx xxxx

[^]	Exclusion Set An exclusion set includes a leading caret, or “hat”, within a set of “square brackets” - [^] An exclusion set matches any single alphanumeric character that is <u>not</u> within the set.

Example:
To match any arbitrarily long sequence of digits that does not start with * we would write our matching rule as follows:
[^*]xx.
In the case of Sydney, Australia, local numbers are 8 digits long, starting with any digit between 3 and 9. We could write this rule as [3-9]xxx xxxx or using an exclusion set we could write:
[^0-2]xxx xxxx

!	! Call Bar To bar users from calling numbers that match a rule, add an “exclamation mark” - ! - in front of that rule in the digit map. The rule is then referred to as a *barring rule*.

Example:
If we wish to bar all calls to 1900 numbers (regardless of length) we could use the following rule:
!1900xx.

< *elements* : *literals* >	Element to Literal Transformation An element-to-literal transformation allows us to substitute the digit sequence matching *elements* with the given *literals*. The expression is contained within a set of “pointy brackets” - < > - and elements are separated from literals using a colon :

Usage Notes:
Single quote ‘ ‘ syntax is NOT needed or allowed for the *literals* in this context; special characters may be used here as they do not apply in this context either.
Elements can be empty, in which case the colon - : - may be omitted. This case is useful for inserting some extra digits in certain part of the dialed digits.
The literals part can be empty also but the colon - : - MUST NOT be omitted. This case is useful for removing part of dialed digits. *Elements* and *literals* MUST NOT both be empty.
Example:
This rule can be used to either remove digits from a dialed number, add digits to a dialed number or transform a dialed number. First, let's look how to transform one number into another. In this case we are going to transform the entered digits of 112 to send out 000. To do this we would write:
<112:000> *take 112 and replace it with 000*

Next, let's strip off preceding digits in a number, in this case if a user dials 02 xxxx xxxx let's send on just the xxxx xxxx without the 02:
<02:>xxxx xxxx *take 02, replace it with nothing then match the next 8 digits*

If we want to add to the start of a dialed number, for example to add 02 to the start of an 8-digit number, we would write:
<02>xxxx xxxx *or* <:02>xxxx xxxx

S, S0, S1 … S9	Digit Timer

	The digit timer should only be used either as the first element of a rule (for a hot or warm line implementation) or as the last element of a rule as a means of overriding the default inter-digit timer. The digit timer – S – is CASE SENSITIVE.
Example: The notation S0, S1, S2, S9 gives digit timer values of 0, 1, 2 and 9 seconds respectively; S is equivalent to S1; S0 is the same as "blank". You can concatenate multiple S elements together if you need more than 9s timeout, such as S9S5 for a 14s timeout. To create a hotline to the number 1-408-890-6000 we would write: <S0:14088906000>	

The next two elements, (*map*) and (M*label*), imply that the OBi digit maps are <u>*recursive*</u>. Recursive digit maps allow digit maps to be reused and make their specification more compact and readable. It is important that you do not specify digit maps that lead to infinite recursion. For example, *a digit map must not include a named embedded digit map that references itself*.

(*map*)	Embedded Digit Map An embedded digit map contains elements within a pair of brackets – ( ) – that are used to match subsequent digits.
Example: To match any number that starts with *74, followed by 1 or 2 digits we would write our matching rule as follows: *74(x|xx)	

(M*label*)	Named Embedded Digit Map A named embedded digit map is used for matching subsequent digits, where *label* refers to one of abbreviated terminal names or a user defined digit map label. As an example, (Msp1) refers directly to the Digit Map contained within *Voice Services* → *SP1 Service* of the device admin webpage.
Example: Let's consider one of the OBi's default digit map rules that directs a call preceeded by **2 to SP2. We want to take all the digits after **2 to be matched with SP2's digit map. To do this we write our rule as: **2(Msp2)	

The following advanced elements allow you to further compact your rules for more elegant digit map syntax:

X	X Wildcard The “uppercase x” – X – wildcard matches any digit from 0-9 as well as * making it useful for use digit maps that include star codes. It is important to note that X is CASE SENSITIVE.
Usage Notes: X is equivalent to [x*] or [0-9*x] *Example:* If we wish to catch all three digit numbers including star codes, we could write our matching rule as follows: XXX	

@	@ Wildcard The “at symbol” - @ - wildcard matches <u>any</u> alphanumeric character except #
Example: To match any arbitrarily long alphanumeric sequence (except #) that does not start with * we would write our matching rule as follows: [^*]@@.	

?	? Matching Function The “question mark” - ? - matching function matches 0 or 1 x, X or @ This function can be used to capture a string of entered numbers or characters of multiple known fixed lengths
Example: The expression xxxx? would catch any number that is entered that is either 3 or 4 digits in length. Let’s say we live in an area where local numbers vary in length. In this example we will use the Brampton area in the UK that has 4 and 5 digit local numbers. Rather than using two rules in our digit map such as (…|xxxx|xxxxx|…) to match the local numbers, we can write a more elegant matching rule as follows: xxxxx?	

Digit Map Rule Examples

Here are some further examples of digit map rules:

Function	Digit Map Rule
Match any 11-digit number starting with 1-408	1 408 xxx xxxx
Match any 7-digit number and prepend 1408 to the number when making the call	<1408> xxx xxxx *or* <:1408> xxx xxxx
Match any number that starts with 011 followed by one or more digits	011xx.
Add '+' to any 11-digit number that starts with 1	<+>1xxxxxxxxxx
Match any number that starts with **1 020 3, 7 or 8 followed by 7 digits and remove the **1 prefix when making the call	<**1:>020 [378]xxx xxxx
Create a hotline to 1234	<:1234> *or* <S0:1234>
Create a warm line to 1234 that will be called if the user doesn't enter any digits within 4 seconds	<S4:1234>
Match any number with at least 8 digits that ends with 8537683, such as 1 510 853 7683, 9 853 7683	xx.853 7683
Match any number with at least 10 digits that ends in 408-890-6000, such as 1 408 890 6000, 001 408 890 6000, +1 408 890 6000	@. 408 890 6000
Add a # to the end of any number with 1 or more digits	xx.<#>
Bar calls to premium rate numbers beginning with 084, 087 and 09	!08[47]x. *and* !09x.

Now we will create <u>an example</u> digit map using a few of the rules above. One important function of a digit map is to determine if the user has entered sufficient digits during dialing, given the array of number combinations available, a digit map normally contains more than one rule. To create a digit map that includes a few of the example rules in the table, we write our digit map contained with brackets ( ) and with each rule separated by a | bar as follows:

(<1408> xxx xxxx | @. 408 890 6000 | xx.<#> | !08[47]x. | !09x.)

Note that spaces have been used in the digit map. It is fine to include spaces to help make your digit map more readable.

Matching Against Multiple Rules in Digit Map

One important function of a digit map is to determine if sufficient digits have been entered by the user during dialing. A digit map normally contains more than one rule. The Digit Map Processor (DMP) must return the best matched rule at some point, or declare the input digit sequence is invalid. The DMP keeps refining its decision as each digit is entered until it reaches a *final decision*, or will be forced to make a *timely decision* when the interdigit timer expires.

The DMP restarts the interdigit timer on every newly entered digit. The duration of this timer can be either *long* or *short*. The long and the short timer values are set to 10s and 2s respectively by default and are configurable under the ***Phone Settings*** group via the **DigitMapLongTimer** and **DigitMapShortTimer** parameters respectively. Whether to use the long or short interdigit timer depends on the current rule matching states. The DMP maintains a matching state for each rule in the digit map as it processes each input digit. The following states are defined:

- Partially Matched (PM) – The rule partially matches the accumulated input sequence. Initially all rules are in this state before any digit is entered. Rules in this state have the potential of becoming EM or IM as more digits are entered. Example: 1234 partially matches the rules xxxxxxx, 1xxxx, 1234567, <123:>xxxx.
- Exactly Matched (EM) – The rule exactly matches the accumulated input sequence. However, any further input digit will turn this rule into the MM state. Example: 1234 exactly matches the rules xxxx, 1234, 1xxx, <123:5678>x
- Indefinitely Matched (IM) – The rule matches the accumulated input sequence indefinitely, with a variable length such that the rule can potentially stay as IM as more matching digits are entered. Example: 011853 indefinitely matches the rules xx., 011xx., <011:>xx.
- Mismatch (MM) – The rule does not match the accumulated input sequence. This state will not change as more digits are entered. Example: 1234 mismatches the rules 123, 1xx, 12345

Rules in the EM or IM state are candidates to be selected by the DMP. After processing a new digit, the DMP returns a final decision if any of the following conditions holds:

1. All rules are the MM state. DMP returns an error
2. One or more rules are in the EM state with no rules in the IM state. DMP returns the best matched EM rule. If the best matched rule is a barring rule, DMP returns an error instead

Otherwise, the DMP starts the short interdigit timer if there is at least one rule in the EM state, or else the long one. When the interdigit timer expires, the DMP makes a timely decision by returning the best matched rule at that moment if one is found, or else a timeout error. Again if the best matched rule in this case is a barring rule, the DMP returns an error instead. Note that the timer to wait for the first input digit is NOT governed by the interdigit timer, but the duration of dial tone being played and could be a lot lengthier than the long interdigit timer.

The best-matched rule is the one that has the most specific literals matching the input digit sequence. For example, the input sequence 1234 matches the rule 123x better than 1xxx. On the other hand, an EM rule is always selected over an IM rule.

Finally, the default interdigit timer can be overridden by appending the S*n* element at the end of the rule where *n* = 0–9 designating the number of seconds to wait before triggering the rule with a timer event.

Here are some more examples:

Consider the simple digit map `(<1408>xxx xxxx)`. As soon as 7 digits are entered, the DMP returns a complete number by prepending the accumulated digits with `1408`.

Consider another simple map `(xx.)`. After the user dials one or more digits, the DMP returns the accumulated digits as a complete number when the long interdigit timer expires.

If we combine the last two maps into one: `(xx. | <1408>xxx xxxx)`. After the user dials 1 or more digits but less than 7 digits, the DMP would return the accumulated digits as a complete number when the (long) interdigit timer expires. As soon as 7 digits are entered, the DMP would return 1408 followed by the accumulated 7-digit when the (short) interdigit expires. On the 8th digit and beyond, however, the DMP will consider the first rule only and return the accumulated digits as is when the (long) interdigit timer expires.

Now add a S4 timer to the second rule: `(xx. | <1408>xxx xxxxS4)`. In this case the DMP behaves exactly the same as the last, except that the short interdigit timer the DMP uses upon receiving the 7th digit is overridden by a 4s timer; hence the user will have up to 4s instead of 2 to dial the 8th digit.

Forcing Interdigit Timeout With The Hash/Pound (#) Key

When dialing, user may force an interdigit timeout with a # key instead of waiting for the DMP to timeout its own long or short timer. This is allowed as long as the # key does not match the current element of any PM rules. Otherwise the # key will be consumed by the DMP instead of triggering a timeout.

Consider the digit map `(33xx.)`. If the user enters 333#, the DMP will return immediately with the number 333.

Now consider the digit map `(33xx.|333#1234x.)`. If the user enters 333#, the DMP will not return but continue to wait for further input or its interdigit timer to expire. Note that the first rule "33xx." is now in the MM state since the digit # does not match "x". The user may continue to enter 1234#, or 1234 and wait for a long interdigit timeout for the DMP to successfully return 333#1234.

Invoke Second Dial Tone in Digit Map

You can tell the OBi to start a tone after a certain pattern of digits have been dialed by specifying the element {t=<*tone*>} within a digit map, where <*tone*> is a 1 to 3-letter name of the tone to play. The tone will stop when the next digit is entered. For example:

(**1{t=di2}(Msp)|**8{t=od}(Mbt))

tells the device to play Second Dial Tone when **1 is dialed, or play Outside Dial Tone when **8 is dialed.

Here is a full list of acceptable (case insensitive) values of <tone>:

bu = Busy Tone

cf = Call Forwarded Dial Tone

cm = Confirmation Tone

co = Conference Tone

    cw1 – cw10 = Call Waiting Tone 1-10, respectively

di = Dial Tone

    di2 = Second Dial Tone

    fb = Fast Busy Tone

ho = Holding Tone

od = Outside Dial Tone

pr = Prompt Tone

rb = Ringback Tone

ro = Reorder Tone (same as fast busy)

si1 – si4 = SIT TONE 1 – 4, respectively

st = Stutter Tone

0 – 9, *, #, a – d = DTMF 0 – 9, *, #, A – D respectively

Change Interdigit Long Timer Dynamically After Partial Match

The OBi starts off with the interdigit long timer set to the configured **DigitMapLongTimer** value when processing a new digit sequence by a digit map. You may change the long timer as some patterns are partially matched by embedding the syntax {L=<*time*>} within a rule in the digit map, where <*time*> is the desired number of seconds for the long timer.

For example: `(011 853 xxxx xxxx{L=5}x. |xx.)`. Here the long timer is shortened to 5s after the user has entered 011 853 + 8 digits. Hence the OBi will declare that a complete number is collected in 5s when no more digits are received. Without the {L=5} syntax the user will have to wait for 10s (by default) for the same to happen.

User Defined Digit Maps

There are 10 user definable digit maps available under the ***User Settings – User Defined Digit Maps*** section of the device configuration web page. These digit maps are referred to as User Defined Digit Map 1 to 10. Each user defined digit map is specified with 2 parameters:

- **Label**: An arbitrary string for referencing this digit map in other digit map specification. The value should be 2-16 characters long. For example, "friends". In this case, `(Mfriends)` can be referenced in other digit maps, such as ***Phone Settings***::**DigitMap**
- **DigitMap**

By default both parameters are empty, except for User Defined Digit Map 1 (see the section below).

A User Defined Digit Map For IPv4 Dialing

The default values of the parameters for User Defined Digit Map 1 are set the following values to support IPv4 Dialing:

Label = `ipd`

Digit Map = `(xx.<*:@>xx?x?<*:.>xx?x?<*:.>xx?x?<*:.>xx?x?|`

`xx.<*:@>xx?x?<*:.>xx?x?<*:.>xx?x?<*:.>xx?x?<*::>xx?x?x?x?)`

The map (Mipd) is referenced in the default setting of the **DigitMap** in ITSP Profile A and B. It supports the following two forms of IPv4 dialing:

a) <user-id>*<a>*<b>*<c>*<d>

b) <user-id>*<a>*<b>*<c>*<d>*<port>

Where <user-id> is an arbitrary length numeric user-id, such as 100345, <port> is a port number in the range 0–65535, and each of <a>,<b>,<c>,<d> is a 1-3 digit pattern in the range 1–255 that identifies one byte of an IP address. The dialed number will be translated into <user-id>@<a>.<b>.<c>.<d> and <user-id>@<a>.<b>.<c>.<d>:<port> respectively. Here are some examples:

1234*192*168*15*113	maps to 1234@192.168.15.113
123456*192*168*15*180*5061	maps to 123456@192.168.15.180:5061

Administrative Features

This section summarizes the administrative features of the OBi1000.

Native Web Server

A built-in web server on the phone, for viewing and changing parameter values, that is protected by an admin password for admin level access and by a user password for user level access. Admin level has full access to all configuration parameters. The administrator can decide which parameters are hidden, read-only, or read-writable at the user level using parameter attributes in a configuration file.

Syslog

The OBi1000 can be set up to send out syslog messages for trouble-shooting. To capture syslog messages from the OBi10000, you need to run a syslog server application at an IP address x.y.w.z that is reachable from the phone. On the phone side, IP address and listening port of the syslog server are configured in ***Device Admin – Syslog***::**Server** and ***Device Admin – Syslog***::**Port** respectively. The default **port** value is 514.

To include detail SIP messages on an SP service in the syslog, use the parameter ***SPn Service***::**X_SipDebugOption**. with one of the options: `Disable`, `Log All Messages`, or `Log All Except REGISTER Messages`. ***SPn Service***::**X_SipDebugExclusion** is a list of SIP methods (request and responses) to exclude from the log. For trouble-shooting a call flow for example, methods such as OPTIONS that are used for keep-alive purpose may be excluded from the log in most cases.

If you do not have a syslog server software application, you may download one here.

Factory Reset

Resetting all configuration parameter to factory default values, or to the customized default values for phones that are customized with some non-generic parameter values

Firmware Update

Automated Firmware Update

Set up rules in the configuration the automatically check and download new phone firmware from a server

Configuration Backup and Restore

Backing the current configuration to restore it later (or apply it to another phone)

Auto Provisioning

OBiTALK Provisioning

Using OBiTALK portal as the provisioning server

ITSP Provisioning

Using the ITSP's provisioning system for provisioning the phones

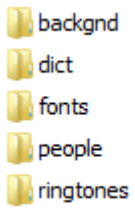
Customization Data Auto Update

OBi Phone can be customized in many ways and some of customization features require extra data such as background pictures, dictionary, fonts etc. The customization data is categorized into two levels: Service Provider (ITSP) Level and End User (User) Level. The data for each level is allocated at dedicated location inside of internal storage in the phone. Customization data auto update only manages the data for ITSP level

Customization Data Package

OBi Phone can only accept data in the form of a tar file which called Phone Customization Data Package. This tar file can be either gzipped or not gzipped. The size of tar file must be smaller than 30 MBytes.

Inside tar file, the data should be organized in the following directory hierarchy, as OBiPhone will search certain data from the certain subdirectory. For example, the power-up logo must be named as "logo.raw" and is allocated under subdirectory "backgnd".



Example Linux command to generate a package named "itsp20141001.tar.gz" assuming the above directory hierarchy is under a working directory called "itspdata". Note that the command must be issued inside of this working directory ("itspdata"), as OBi Phone will NOT strip the first level of the tarball.

% tar –zcvf ../itsp20141001.tar.gz *

Example Linux command to generate the MD5 checksum:

% md5sum itsp20141001.tar.gz

Auto Update Operation

Similar to Auto Firmware Update feature, Customization Data Auto Update is configured by a set of parameters. When it is enabled, OBi phone attempts to download the Customization Data Package according to the scripts specified in the "DownloadURL", then the data of the package will be validated by the given MD5 checksum before it is installed to internal storage. As soon as data installation is completed successfully, OBi Phone will restart itself (warm boot) and will not try to download the package again until the MD5 checksum is changed, even if the feature is still enabled.

Auto Update Operation is always performed after firmware update and all configuration provisioning are completed.

Auto Update Configuration Parameters

This feature is configured by the following parameters.

Parameter	Description	Default Setting
Method	Current operational method of Provisioning. Available choices are: - Disabled = Do not download from DownloadURL - System Start = Download from DownloadURL just once on system start - Periodically = Download from DownloadURL on system start, and then periodically at the interval specified in the Interval parameter Note: First download on system start will be performed after firmware update and configuration provisioning are complete	Disable

Interval	When Method is set to Periodically, this is the number of seconds between download from DowloadURL. If value is 0, device downloads once only on system start (i.e., equivalent to setting Method to System Start)	0
DownloadURL	URL of Customization Data Package	
MD5Checksum	Standard MD5 checksum (hexadecimal string) of the Customization Data Package	
Incremental	When enable the customization Data package will be installed incrementally, (e.g) without erasing the old data.	false
DnsLookupType	Choices are: - A Record Only - SRV Record Only - Try Both	A Record Only
DnsSrvPrefix	Choices are: - No Prefix - With Prefix - Try Both	No Prefix
Username	Optional Username for authentication if URL scheme is http://	
Password	Optional Password for authentication if URL scheme is http://	

Device Web Page and Configuration Parameter Reference

This is a reference section that lists all the configuration and status parameters. Status parameters are read only and are not provisionable. Status parameters are highlighted with a different text color.

Status

System Status

System Status is available on the phone built-in portal and on OBiTALK device management portal, under the web page with the same title.

Status Parameter	Description	Example Value
WAN Status (DeviceInfo.Network.Status.WAN.)		
AddressingType	Method currently used by the device to get an IP address assignment	DHCP
IPAddress	IP address currently assigned to the device	192.168.15.165
SubnetMask		255.255.255.0
DefaultGateway		192.168.15.1
DNSServer1		4.2.2.2
DNSServer2		
MACAddress	MAC address installed on the device	9CADEF90004E
WiFi Status (DeviceInfo.Network.Status.WiFi.)		
AddressingType		
IPAddress		
SubnetMask		
DefaultGateway		
DNSServer1		
DNSServer2		
MACAddress		
Product Information (DeviceInfo.)		
ModelName		OBi508
MACAddress		9CADEF90004E
SerialNumber		88H01NA00ZXV
OBiNumber		552 860 300
HardwareVersion		1.1
SoftwareVersion		4.0.1(Build: 4256)
SystemTime	Shows the current time on the system	15:32:35 01/29/2014, Wednesday
UpTime	With last **Reboot Reason** in parentheses. See the list below this table for a list of reboot reason codes	20 Days 5:04:13 (2)
CertificateStatus	Indicate if a device certificate is installed on the unit	Installed
CustomizationStatus	Indicate if this device is a customized unit	Generic
OBiBT Dongle Status (VoiceService.1.X_BT.1.Stats.)		
Status		

Discoverable		
CallState		0 Active Calls
SP*n* Service Status (VoiceService.1.VoiceProfile.1.Line.*n*.), *n* = 1 – 6		
Status	Registration status of this service. If there are problems with the registration or authentication, the SIP 4xx – 6xx error code and error message will be displayed here. This is very useful information for troubleshooting issues with SIP-based services.	Registered (server=192.168.15.118; expire in 39s)
PrimaryProxyServer	IP address of the current Primary Proxy Server if proxy server redundancy is enabled on this service	
SecondaryProxyServer	IP address of the current Secondary Proxy Server if proxy server redundancy and secondary registration are both enabled on this service	
CallState		0 Active Calls
OBiTALK Service Status (VoiceService.1.X_P2P.1.Stats.)		
Status	Connection status with the OBiTALK network	Normal (User Mode)
CallState		0 Active Calls

Reboot Reason Codes

0: Reboot on Power Cycle

1: Operating System Reboot

2: Reboot after Firmware Update via provisioning or phone (***6)

3: Reboot after New Profile Invoked

4: Reboot after Parameter Value Change or Firmware has changed and invoked via device web page

5: Reboot after Factory Reset using the OBi device hardware pin

6: New Profile Invoked AND Profile URL Changed

7: Reboot from SIP Notify (Reserved)

8: Reboot from Telephone Port (IVR)

9: Reboot from Webpage - No change in parameter value(s) or firmware

10: Reboot During OBiTALK Signup

11: Reboot During OBiTALK Signup

12: Reboot after DHCP server offers IP, GW-IP and/or Netmask different from what the OBi device is currently using

13: Reboot on Data Networking Link Re-establishment

16: Reboot after DHCP RENEW NAK received

18: Reboot after WAN IP setting(s) changed

30: Reboot from Phone GUI – No change in parameter value(s)

31: Reboot from Phone GUI - After Parameter Value Change.

Call Status Web Page

Call Status of each call is only available during the lifetime of the call. It is removed as soon as the call is ended.

The Call Status page shows a number of running call statistics and state parameters for each active call currently in progress. The following information and statistics are shown for each call:

Status	Description
Peer Name	Call Peer's Name
Peer Number	Call Peer's Number
Start Time	Starting time of the call
Duration	Duration of the call
Peer RTP Address	The peer address:port where RTP packets are sent to
Local RTP Address	The local address and port where RTP packets are sent from
RTP Transport	The transport used for RTP (UDP, TCP, or SSL)
Audio Codec	The audio encoder and decoder being used for this call
RTP Packetization	The transmitted and received packet sizes in milliseconds
RTP Packet Count	Total number of RTP packets transmitted and received thus far
RTP Byte Count	Total number of RTP bytes transmitted and received thus far
Peer Clock Differential Rate	Clock difference between this device and the peer in ppm (parts per million)
Packets Out-of-Order	Number of received RTP packets that are out of order
Packets Lost	Number of incoming RTP packets assumed lost
Packet Loss Rate	Amount of incoming RTP packets assumed lost rate in percent
Packet Drop Rate	Amount of incoming RTP packets dropped in percent
Jitter Buffer Length	Size of the current jitter buffer in milliseconds
Received Interarrival Jitter	Average measured network jitter in the received direction in milliseconds
Max Interarrival Jitter	Maximum measured network jitter in the received direction in milliseconds
Jitter Buffer Underruns	Amount of jitter buffer underruns during the call

For each entry on the call status page, the following buttons may be available:

- **Remove**: This button is available for all calls. Pressing this button will end that call.
- **Record**: This button is available for calls involving the Phone port only. Pressing this button allows you to record the current conversation in an audio (.au) file

SP Services Statistics

This page shows the following information for each SP*n* service, where *n* = 1 – 6.

Parameter	Description	Default Setting
Reset Statistics (VoiceService.1.VoiceProfile.1.Line.*n*.Stats.), *n* = 1 – 6		
ResetStatistics	Check this option and press "submit" to reset the statistics for this SP Service	NA
RTP Statistics (VoiceService.1.VoiceProfile.1.Line.*n*.Stats.), *n* = 1 – 6		
PacketsSent	Total RTP packets sent on this line	NA
PacketsReceived	Total RTP packets received on this line	NA
BytesSent	RTP payload bytes sent for this line	NA
BytesReceived	RTP payload bytes received for this line	NA
PacketsLost	Number of RTP packets lost on this line	NA

Overruns	Number of times receive jitter buffer overrun on this line	NA
Underruns	Number of times receive jitter buffer underrun on this line	NA

OBiWiFi Configuration Web Page and Parameter Reference

WiFi Settings Web Page

This page shows all the WiFi setup parameters and status:

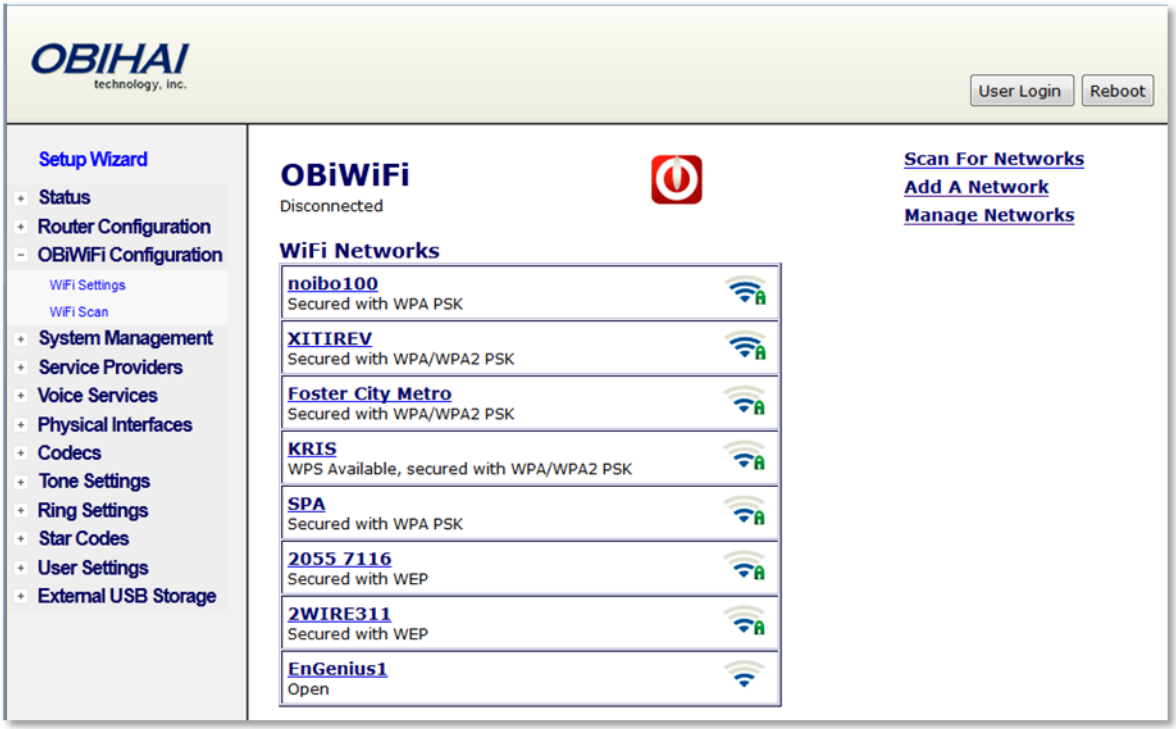
Parameter	Description	Default Setting
Basic Settings (DeviceInfo.WiFi.Basic.)		
Enable	Enable OBiWiFi feature. You must have an OBiWiFi dongle attached to the OBi to use the feature	true
PreferredAccessPoint	Indicate which access point to use when more than one remembered AP are in range. Select from the list: None, Access Point 1, Access Point 2, …, Access Point 20. This value is automatically populated with the last AP that OBi user chose to connect explicitly from the device web page	None
ShowAccessPointPassword	Check this box and press submit to show all the AP passwords in (unmasked) plain text (no reboot required). The passwords will be masked again following a reboot of the device	false
Internet Settings (DeviceInfo.WiFi.)		
AddressingType	The method to assign an IP address to this interface. Choose between DHCP or Static	DHCP
IPAddress	The IP address to use if AddressingType = Static	
SubnetMask	The subnet mask to use if AddressingType = Static	
DefaultGateway	The default gateway to use if AddressType = Static	
DNSServer1	An additional DNS Server to use in addition to the ones received from DHCP	
DNSServer2	An additional DNS Server to use in addition to the ones received from DHCP	
Access Point n (DeviceInfo.WiFi.AP.n.), n = 1, 2, …, 20		
SSID	SSID of the access point	
Password	Password or pass-phrase based on the authentication method used by the AP. For WPA, the pass-phrase should be no more than 64 characters. For WEP, the password should be in one of the four formats: 10 HEX digits, 26 HEX digits, 5 ASCII characters, or 13 ASCII characters. The HEX digits can be upper or lower case	
SecurityEnabled	This is a read only parameter. It indicates if the AP has security enabled or not	

WiFi Scan Web Page

The WiFi Scan device page offers a familiar user interface to let you scan for access points in the neighborhood. A screenshot of this page is shown below. You can click on the page one of the available AP to connect to. If the AP requires authentication but the OBi does not have any valid credential, a page will be returned to prompt you to enter a password or pass-phrase and press “Connect” to continue.

If your AP does not show up as a listed device on this page, e.g. perhaps its SSID is not broadcast, you may enter its SSID and security credentials manually by clicking the “Add a Network” link. The “Manage Networks” link takes you back to

the WiFi Settings device page, whereas the “Scan For Networks” link reloads this page in order to rescan for the access points in the neighborhood.



System Management Parameters

WAN Parameters

Available at the WAN Settings Page

Parameter	Description	Default Setting
Internet Settings (DeviceInfo.WAN.)		
AddressingType	The method used for assigning IP address, subnet mask, default gateway, etc., to the device. Available choices are: DHCP: IP address, default gateway, etc. are assigned by DHCP Server Static: IP address, default gateway, etc. are taken from the manually configured values PPPoE: IP address default gateway, etc. are acquired by PPPoE Protocol (OBi202 only)	DHCP
IPAddress	The IP address to assign to the device when AddressingType is set to Static	
SubnetMask	The subnet mask to use when AddressingType is set to Static	
DefaultGateway	The default gateway IP address to assign to the device when AddressingType is set to Static	
DNSServer1	IP address of the first DNS server to use, in addition to the ones obtained from the DHCP server when DHCP is also enabled. If AddressingType is set to Static, the device only uses DNSServer1 and DNSServer2 for DNS lookup. It will try up to 5 DNS servers when attempting to resolve a domain name. DNSServer1 and DNSServer2 will be tried first, whichever is specified, and then the ones obtained from the DHCP Server if available	
DNSServer2	IP address of the second DNS server to use, in addition to the ones obtained	

	from the DHCP server when DHCP is also enabled. If AddressingType is set to Static, the device only uses DNSServer1 and DNSServer2 for DNS lookup. It will try up to 5 DNS servers when attempting to resolve a domain name. DNSServer1 and DNSServer2 will be tried first, whichever is specified, and then the ones obtained from the DHCP Server if available	
MACAddressClone	The MAC Address to clone (instead of using the factory installed MAC Address)	
PPPoEACName	PPPoE access concentrator name. Enter if it is required	
PPPoEServiceName	PPPoE service name. Enter if it is required	
PPPoEUsername	PPPoE account username provided by your ISP	
PPPoEPassword	PPPoE account password	
PPPoEKeepAlive	PPPoE keep alive period in seconds	60
VLANEnable	Enable VLAN Operation	false
VLANID	Valid range is 0 – 4094 (4095 is reserved). 0 means VLAN is disabled and egress packets are not tagged by the device. This setting applies to all packets sent by the device	0
VLANPriority	Valid choices are 0 – 7. This setting applies to all packets sent by the device.	0
LLDP-MED	Enable LLDP-MED discovery	false
LLDP-MEDExclusivePeriod	Number of seconds for LLDP-MED discovering Network Policy exclusively before the IP is established according to AddressType.	0
LLDP-MEDAssetID	Customizable AssetID to be included in Inventory Management TLV. The default is OBi Number	$OBN
Local Time (DevicefoInfo.Time.)		
CurrentLocalTime	Current local date and time of the device (read only)	
Time Service Settings (DevicefoInfo.Time.)		
NTPServer1	Hostname or IP address of the first NTP server	pool.ntp.org
NTPServer2	Hostname or IP address of the second NTP server	
LocalTimeZone	Local time zone. Available choices are: - GMT-12:00(Int'l Dateline West) - GMT-11:00(Samoa) - GMT-10:00(Hawaii) - GMT-09:00(Alaska) - GMT-08:00(Pacific Time) - GMT-07:00(Mountain Time) - GMT-06:00(Central Time) - GMT-05:00(Eastern Time) - GMT-04:00(Atlantic Time) - GMT-03:30(Newfoundland) - GMT-03:00(Buenos Aires,Greenland) - GMT-02:00(Mid-Atlantic) - GMT-01:00 - GMT+00:00(London,Lisbon) - GMT+01:00(Rome,Paris,Madrid) - GMT+02:00(Athens,Cairo)	GMT-08:00

	- GMT+03:00(Moscow,Baghdad) - GMT+04:00(Abu Dhabi) - GMT+04:30(Kabul) - GMT+05:00(Islamabad,Karachi) - GMT+05:30(New Delhi) - GMT+05:45(Kathmandu) - GMT+06:00 - GMT+07:00(Bangkok,Jakarta) - GMT+08:00(Beijing,HK,Singapore) - GMT+09:00(Tokyo,Seoul) - GMT+10:00(Sydney,Guam) - GMT+11:00(Solomon Is.) - GMT+12:00(Fiji,Auckland)	
DaylightSavingTimeEnable	Enable daylight saving time on the unit	true
DaylightSavingTimeStart	Daylight Saving Time Start Date. Format: month/day/weekday/hh:mm::ss, where month=1-12, day=±(1-31), weekday=0,1-7 (0=special, 1=Monday, 7=Sunday), hh=0-23,mm=0-59,ss=0-59. If weekday=0, daylight saving starts on the given month/day; otherwise it starts on the weekday on or after the given month/day if day > 0, or on the weekday on or before the last-day-of-given-month+day+1 (note that day = -1 equivalent to last day of the month). :ss may be omitted if the value is 0; :mm:ss may be omitted if mm and ss are both 0.	3/8/7/2
DaylightSavingTimeEnd	Daylight Saving Time End Date. Same format as Start Date	11/1/7/2
DaylightSavingTimeDiff	Amount of time to add to current time during Daylight Saving Time. Format: [-]hh:mm:ss. :ss may be omitted if it is 0; :mm:ss may be omitted if both are 0.	1
DHCP Client Settings (X_DHCPC.)		
Hostname	Device Hostname to be sent to server	$DM
ExtraOptions	Comma separated list of extra DHCP options to be requested. The supported options are 66, 150, 159, 160, 161	66
DNS Control (X_DNSControl.)		
DNSQueryOrder	Controls the order in querying DNS Servers. The following choices are available: DNS Server1, DNS Server 2, DHCP Offered DNS Servers DHCP Offered DNS Servers, DNS Server1, DNS Server 2 DNS Server1, DNS Server 2	DNS Server1, DNS Server 2, DHCP Offered DNS Servers
DNSQueryDelay	Controls the delay (in seconds) before trying the next DNS server when resolving a domain name. Available choices are: 0, 1, 2, 3, 4, and 5.	2
Local DNS Records (X_LocalDNSRec.)		
N where *N* = 1 – 32	One of 32 Local DNS Records (numbered 1 – 32). Each record is a mini script of the following format: *Name=A,A,A,...* **OR** *Name=R,R,R,...*	

	where *Name* represents the domain name to be resolved locally, and has the format *prefix+domain* (such as `machine.sip+obihai.com`). Everything after '+' is considered as the *domain* to be appended to the *host* field in each *R* on the right hand side. '+' is optional; if missing the full domain must be used in every *R*. *A* represents an A record which is just an ip address, such as 192.168.12.17. *R* represents an SRV record and has the format: {*host*:*port*,*pri*,*wt*} where - *host* is a hostname with or without domain part (such as xyz, xyz.abc.com.). A dot (.) at the end of *host* indicates it is a complete hostname that does not require the domain to be appended. - *port* is a port number (such as 5060) - *pri* is the priority. Valid value is 0(highest) – 65535(lowest) - *wt* is the weight. Valid value is 0(lowest) – 65535(highest) *wt* is optional; 1 is the default if not specified. *pri* is optional only if *wt* is not specified; 1 is the default if not specified. *port* is optional; the default to use will be based on the protocol (5060 for SIP, 80 for HTTP, etc.) . The enclosing curly braces { } are also optional if there is only one *R*; or if there is no comma used inside the *R*. Examples: _sip._udp+obihai.com=abc,xyz,pqr:5080,{mmm,2},{super.abc.com.} abc.obihai.com=192.168.15.118,192.168.15.108 Note: If the A record of a given hostname cannot be found in any of the Local DNS Records, the device will attempt to resolve it using external DNS queries. Any change applied to local DNS Record needs reboot in order to take effect.	

Auto Provisioning Parameters

Available at the Auto Provisioning Page

This page shows all the parameters related to remote provisioning of the device, as shown in the following table. Provisioning is an important topic for deployment by service providers. Please refer to this document for details on OBi Device provisioning (http://obihai.com/docs/OBiProvisioningGuide.pdf)

Parameter	Description	Default Setting
Auto Firmware Update (X_DeviceManagement.FirmwareUpdate.)		
Method	Current operational method of auto firmware updating. Available choices are: - Disabled = Do not check for f/w upgrade from FirmwareURL - System Start = Check for f/w upgrade from FirmwareURL just once on system start - Periodically = Check for f/w upgrade from FirmwareURL on system start, and then periodically at the interval specified in the	Disabled

	Interval parameter - Time Of Day = Download from ConfigURL once a day at the time specified in the TimeofDay parameter Note: First f/w upgrade check on system start will be performed after a random delay of 0-30s	
Interval	When Method is set to Periodically, this is the number of seconds between each checking of f/w upgrade check from FirmwareURL. If value is 0, device checks once only on system start (i.e., equivalent to setting Method to System Start)	0
TimeofDate	The time during the day at which to execute ConfigURL is the Method is set to "Time Of Day". It must be specified in the format: hh:mm[+rr] where: - hh: 0 – 23 - mm: 0 – 59 - +rr: (optional) maximum range of random delay in minutes from 0 – 360. If not specifieid, the default is 30 minutes. Setting rr = 0 will remove the random delay.	00:00+30
RandomDelayRange		30
FirmwareURL	URL of firmware package. URL must include scheme. Supported schemes are http:// and tftp://	
DnsLookupType	Choices are: - A Record Only - SRV Record Only - Try Both	A Record Only
DnsSrvPrefix	Choices are: - No Prefix - With Prefix - Try Both	No Prefix
Username	Optional Username for authentication if URL scheme is http://	
Password	Optional Password for authentication if URL scheme is http://	
ITSP Provisioning (X_DeviceManagement.ITSPProvisioning.) and **OBiTALK Provisioning** (X_DeviceManagement.Provisioning.)		
Method	Current operational method of Provisioning. Available choices are: - Disabled = Do not download from ConfigURL - System Start = Download from ConfigURL just once on system start - Periodically = Download from ConfigURL on system start, and then periodically at the interval specified in the Interval parameter - Time Of Day = Download from ConfigURL once a day at the time specified in the TimeofDay parameter Note: First download on system start will be performed after a random delay of 30 – 90s. If there is a firmware update scheduled at the beginning. Or a random delay of 10- 70s..	System Start
Interval	When Method is set to Periodically, this is the number of seconds between download from ConfigURL. If value is 0, device downloads once only on system start (i.e., equivalent to setting Method to	0

	System Start)	
TimeofDate	The time during the day at which to execute ConfigURL is the Method is set to "Time Of Day". It must be specified in the format: hh:mm[+rr] where: - hh: 0 – 23 - mm: 0 – 59 - +rr: (optional) maximum range of random delay in minutes from 0 – 360. If not specifieid, the default is 30 minutes. Setting rr = 0 will remove the random delay.	00:00+30
ConfigURL	URL of config file	tftp://$DHCPOPT66/$MAC.xml Note: Default value is blank for OBiTALK Provisioning
DnsLookupType	Choices are: - A Record Only - SRV Record Only - Try Both	A Record Only
DnsSrvPrefix	Choices are: - No Prefix - With Prefix - Try Both	No Prefix
GPRM0 to GPRM7	Non-volatile generic parameters which can be referenced in other parameters, such as ConfigURL	
TPRM0 to TPRM3	Temporary variables used in scripts for ConfigURL. Please refer to OBi Device Provisioning Guide for examples on how to use these variables	
User Defined Macro *n* (X_DeviceManagement.X_UserDefinedMacro.*n* .), *n* = 0, 1, 2, 3 **($UDM0 – $UDM3)**		
Value	Any plain text, or reference to another parameter's full TR-104 name prepended by a '$'	
ExpandIn	A comma separated list of parameters that are allowed to use this macro expansion. Each parameter must be specified using its full TR-104 name. Specify ANY to allow it in all parameters.	
SyntaxCheckResult	This is read only status value regarding the syntax of the UDM. "Pass" means that this UDM is valid. Otherwise, it shows the syntax error detected by the device either in the Value or ExpandIn parameters of the UDM.	

$MACRO Expansion Supported by the OBi Device

Macro Name	*Description*	*Where It Can Be Used*
MAC	Device MAC address in upper case, such as 9CADEF000000	ANY
MACC	Device MAC address in upper case with colon, such as 9C:AD:EF:00:00:00	ANY
mac	Device MAC address lower case, such as 9c:adef000000	ANY

macc	Device MAC address lower case, with colon, such as 9c:ad:ef:00:00:00	ANY
FWV	Firmware version, such as 1.0.3.1626	ANY
HWV	Hardware version, such as 2.8	ANY
IPA	Device current IP Address, such as 192.168.15.100	ANY
DM	Device Model Name, such as OBi110	ANY
DMN	Device Model Number, such as 110	ANY
OBN	Device OBi Number, such as 200123456	ANY
DSN	Device S/N, such as 88B01NA00000	ANY
GPRM*n* *n*=0–7	Value Auto Provisioning::GPRM*n*	Auto Provining::ConfigURL, Auto Firmware Update::FirmwareURL
TPRM*n* *n*=0-3	Value Auto Provisioning::TPRM*n*	Auto Provining::ConfigURL, Auto Firmware Update::FirmwareURL
UDM*n*, *n*=0-3	Value of User Define Macro *n*::Value	The value of User Define Macro *n*::ExpandIn
DHCPOPT66	Value of DHCP option 66 assigned by server	ANY
DHCPOPT150	Value of DHCP option 150 assigned by server	ANY
DHCPOPT159	Value of DHCP option 159 assigned by server	ANY
DHCPOPT160	Value of DHCP option 160 assigned by server	ANY
DHCPOPT161	Value of DHCP option 161 assigned by server	ANY

Zero-Touch, Massive Scale Remote Provisioning:

OBi ZT or Zero Touch provisioning is a system level approach to deploying and maintaining thousands or millions of OBi devices with high security and control at the device level down to the individual parameter provisioned on each device. Please contact sales@obihai.com for information regarding the capability, process and practice of using OBi ZT Provisioning.

Device Admin Parameters

Available under the Device Admin Page

This page includes the following configuration parameters.

Parameter	Description	Default Setting
Web Server (X_DeviceManagement.WebServer.)		
Port	Web Server Port Number	80
AdminPassword	Administrator Password, case sensitive	admin
UserPassword	User Password, case sensitive	user
LCDScreenShot	Allow user to create LCD Screenshot from the option in Device Update page	false
IVR (X_DeviceManagement.IVR.)		
Enable	Enable IVR for local configuration	true
Password	IVR access password (must be all digits)	

Syslog (X_DeviceManagement.Syslog.)		
Server	IP address of the Syslog Server where the device sends syslog debug messages to. If the value is blank, syslog is disabled	
Port	Syslog Server Port Number	514
Level	Syslog Message Level	7
TAG	An arbitrary string to add to the beginning of each syslog message	
HTTP Client (X_DeviceManagement.HTTPClient.)		
UserAgent	Value of the User-Agent header in all HTTP Requests which are used in firmware upgrade and auto provisioning.	$DM
TimeOut	Timeout in seconds for an HTTP transaction to complete	600

Device Update Web Page

There are five different functions offered on this page. These functions are described below.

Firmware Update

You may upgrade the firmware for your OBi device from the device configuration web page. The firmware file with which you want to upgrade the device must be stored locally on a computer from which you can access with a web browser.

Follow these steps to upgrade:

Step 1: Select the, "System Management – Device Update" menu on the side panel of the web page.

Step 2: Specify the path of the firmware file by clicking the, "Select file to upgrade firmware" box or pressing the, "Browse" button in the Firmware Update section of the page. This will present a file browser window where you can navigate to and select the firmware file.

Step 3: Upon selection of the firmware file, press the "Update" button to start the upgrade process.

The entire process will take about 30 seconds to complete. Note that you MUST NOT disconnect the power from the device during this procedure. If the new firmware is upgraded successfully, the OBi device will reboot automatically to start running the new firmware. Otherwise the page will show an error message explaining why upgrade has failed.

Possible Error Messages on Firmware Update Failure:

Error Message	Description	Suggested Solution
Firmware Package Checksum Error	A corrupted Firmware package file has been used for the update.	Check the file and / or re-download the firmware package and try again.
System Is Busy	The OBi device is busy because one of the phone services is in an active call or device provisioning is in progress.	Try to update again later
Firmware Is Not Modified	The OBi device is already running the same firmware as the one selected for update.	No need to upgrade.

Backup (Customized) AA User Prompts

Up to 20 individual prompts may be recorded through the device IVR interface (see ***Telephone-IVR-Based Local Configuration*** section). These prompts may be backed up into a single file from the web browser. The default name of the file is "backupaa.dat". The backup file also includes the annotations entered for each recorded prompt.

To restore an AA prompt file onto an OBi, do it exactly like a firmware upgrade via the web browser but provide the device with the prompt file instead of a firmware file. The OBi can detect from the file header that you are trying to upload a prompt file and process the file accordingly. *Warning: All the existing prompts in the device will be removed first when applying the backup file; this process cannot be undone*.

Backup Configuration

The current configuration of the OBi device can be backed up and stored as a file in XML format at a user specified location. The default name of the file is "backup*xxxxxxxxxxxx*.xml", where the *xxxxxxxxxxxx*x represents the MAC address of unit.

When backing up a device's configuration, you may select the following three options before selection of the "Backup".

Option	Description	Default Setting
Incl. Running Status	If checked, the value of all status parameters will be included in backup file. Otherwise, status parameters are excluded from the backup	No
Incl. Default Value	If checked, the default value of parameters will be included in the backup file. Otherwise, default values are excluded from the backup	No
Use OBi Version	If not checked, the backup file uses XML tags that are compliant with TR-104 standard. Otherwise, the backup file will be stored in an OBi proprietary format where the XML tags are not compliant with TR-104; but the file size will be smaller and the file will be more readable	No

When the file browser window pops up for, you can change the filename and choose the location to save the backup file. Note that different web browser might handle this differently. If the operation is blocked due to the security setting of the web browser, you should change the security setting temporarily to allow this operation to complete.

Restore Configuration

When restoring the configuration to a previous backup copy, you will need to specify the backup file you want to restore to by selecting the "Browse" button in the Restore Configuration section of the web page. Then, select the "Restore" button to start the process. The OBi device will automatically reboot, after the restoration is complete.

IMPORTANT Note: All passwords and PINs are excluded from the backup file. Hence they will not be available to restore. Call history is excluded from the backup, but can be saved as an XML formatted file separately from the Call History web page.

Reset Configuration (to Factory Default)

The OBi device may be reset to factory default condition. Call history and various statistical information will be removed at the same time. Resetting the device configuration should be used with **extreme caution** as the operation cannot be undone. To do this you press the "Reset" button in the Reset Configuration section. A confirmation window will pop up. The OBi device then proceeds to reset the configuration once you confirm that this is indeed what you want to do. The OBi device will reboot automatically when factory reset is completed.

Service Provider Configuration Parameters

ITSP Profile *X* – General Web Page (*X* = A, B, C, D, E, F)

The following configuration parameters are available on this page.

Parameter	Description	Default Setting

General ITSP Settings (VoiceService.1.VoiceProfile.k.), k = 1 – 6 for X = A – F, respectively		
Name	Human-readable string to identify the profile instance. Maximum Length = 127 characters	
SignalingProtocol	Choose among the following list of signaling protocols for this ITSP: SIP Google Voice	SIP
DTMFMethod	Method to pass DTMF digits to peer device. Available choices are: Inband - DTMF tone are sent as inband audio signal RFC2833 - DTMF tone events are relayed per RFC2833 SIPInfo - DTMF tones are relayed with SIP INFO request Auto - Method to use based on call setup negotiation (either Inband or RFC2833 may be negotiated)	Auto
InbandDTMFVolume	DTMF tone volume when sending inband DTMF. Valid values are -24dB to 0dB in 3dB steps.	-15dB
X_UseFixedDurationRFC2833DTMF	When relaying DTMF digit events on this trunk using RFC2833, the RFC2833 RTP packets normally will keep streaming for as long as the digit is pressed. With this option set to TRUE, the device sends only one RTP digit event packet with a fixed duration of 150 ms regardless how long the digit has been pressed	FALSE
DigitMap	A Digit map to restrict the numbers that be dialed or called with this service. See *OBi Call Routing and Digit Map Section* for a description of digit map syntaxes. Maximum Length = 511 characters	(1xxxxxxxxxx|<1>[2-9]xxxxxxxxx|011xx.|xx.)
STUNEnable	Enable device to send a STUN binding request for its RTP port prior to every call	false
STUNServer	IP address of domain name of the STUN Server to use	
X_STUNServerPort	UDP listen port of the STUN Server.	3478
X_ICEEnable	Enable device to use ICE algorithm to find the best peer RTP address to forward RTP traffic for every call	false
X_SymmetricRTPEnable	Enable device to apply symmetric RTP behavior on every call: That is, send RTP to peer at the address where incoming RTP packets are received from	false
Service Provider Info (VoiceService.1.VoiceProfile.k.ServiceProviderInfo.) , k = 1 – 6 for X = A – F, respectively		
Name	Human-readable string identifying this service provider. Maximum Length = 127 characters	
URL	Website of this service provider. Maximum Length = 127 characters	
ContactPhoneNumber	Phone number to contact this service provider. Maximum Length = 31 characters	
EmailAddress	Email address to contact this service provider. Maximum Length = 127 characters.	

ITSP Profile X – SIP Web Page (X = A, B, C, D, E, F)

The following configuration parameters are available on this page.

Parameter	Description	Default Setting
SIP (VoiceService.1.VoiceProfile.k.SIP.), k = 1 – 6 for X = A – F, respectively		
ProxyServer	Host name or IP address of the SIP proxy server	
ProxyServerPort	Destination port to connect to the SIP server	5060
ProxyServerTransport	Transport protocol to connect to SIP server. The three choices are UDP, TCP, or TLS	UDP
RegistrarServer	Hostname or IP address of the SIP registrar. If a value is specified, device sends REGISTER to the given server; otherwise REGISTER is sent to *ProxyServer*	
RegistrarServerPort	Destination port to connect to SIP registrar	5060
RegistrarServerTransport	Transport protocol to connect to registrar. This parameter is reserved for future. The only choice is *UDP*	UDP
UserAgentDomain	CPE domain string. If empty, device uses ProxyServer as its own domain to form its AOR (Address Of Record) or Public Address when constructing SIP messages (for example, in the FROM header of outbound SIP Requests). Note: If **`SPx Service::`**`URI` is specified, additional rules applied in forming the AOR. See description of `URI` parameter for more details and examples	
OutboundProxy	Host name or IP address of the outbound proxy. Outbound proxying is disabled if this parameter is blank.	
OutboundProxyPort	Destination port to be used in connecting to the outbound proxy	5060
X_OutboundProxyTransport	A different SIP transport may be used by the OutboundProxy. The available choices are: UDP TCP TLS Follow ProxyServerTransport	Follow ProxyServerTransport
X_BypassOutboundProxyInCall		false
RegistrationPeriod	Nominal interval between device register in seconds	60
X_RegistrationMargin	Specifies the margin to renew SIP registration before it expires. The default is to renew at half-time before the next expiration if the expires value is less than 1200s; or 600s before expiration otherwise. You can specify here the number of seconds before expiration to renew registration explicitly (such as 10 or 20), or as a fraction of the current register expires value (such as 0.1 or 0.25)	
TimerT1	Value of SIP timer T1 in ms	500
TimerT2	Value of SIP timer T2 in ms	4000
TimerT4	Value of SIP timer T4 in ms	5000
TimerA	Value of SIP timer A in ms	500
TimerB	Value of SIP timer B in ms	32000
TimerD	Value of SIP timer D in ms	32000
TimerE	Value of SIP timer E in ms	500
TimerF	Value of SIP timer F in ms	32000

TimerG	Value of SIP timer G in ms	500
TimerH	Value of SIP timer H in ms	32000
TimerI	Value of SIP timer I in ms	5000
TimerJ	Value of SIP timer J in ms	32000
TimerK	Value of SIP timer K in ms	5000
InviteExpires	Invite request Expires header value in seconds	60
ReInviteExpires	Re-invite Expires header value in seconds	10
RegisterExpires	Register Expires header value in seconds (not used at the moment)	3600
RegistersMinExpires	Register Min-Expires header value in seconds (not used at the moment)	15
RegisterRetryInterval	Register retry interval in seconds	30
X_RegisterRetryResponseCodes	A set of rules to control how many seconds to wait before retrying register after a specific failure response. The rules are specified with a digitmap string; the value after the letter w designates the number of seconds to wait. A range may be specified with two numbers with a – in between such that a random delay within that range is used.	(<40[17]:w120>|<40[34]:w120> |<99[01]:w120-200>|[4-9]xx)
X_RegisterIncludeInstance	Enable/Disable inclusion of instance parameter in the Contact of REGISTER requests.	true
DSCPMark	Diffserv code outgoing SIP packets	0
X_SpoofCallerID	Allow outbound Caller ID spoofing. If set to Yes, device will attempt to set the caller-id name and userid field in the FROM header to that of a remote caller in the case of a bridged call (from another trunk, such as PSTN Line or another SP Service). Otherwise, device always its own account information to form the FROM header. Note that most service provider will not allow originating a call if the FROM header field does not match the account credentials. Enable this option only if you are sure that the service provider allows it. For example, an IP PBX may allow it.	false
X_UseRefer	Enable the use of SIP REFER for call transfer. If disabled, device will bridge the call instead when performing a call transfer (which consume some resources on the device)	false
X_ReferAOR	Enable the use the target's AOR (Address of Record or public address) in Refer-To header of SIP REFER. If disabled, the target's Contact will be used instead	true
X_Use302ToCallForward	Enable the use of 302 response to INVITE for call forward. If disabled, device will bridge the call legs instead when forwarding a call (and will consume some resources on the device)	true
X_UserAgentName	If a value is specified, device includes a User-Agent header in all SIP Requests, or a Server header in all SIP responses, that contains exactly the given value	OBIHAI/${DM}-${FWV}
X_ProcessDateHeader	Enable the device to decode the DATE header sent by the ITSP in a 200 response to its REGISTER. The DATE header specifies the current GMT time and the device can use to	true

	adjust its local time and date without relying on NTP	
X_InsertRemotePartyID	Enable the device to include a Remote-Party-ID header in its outbound SIP INVITE to indicate to the ITSP the caller's preferred privacy setting (either full or none)	true
X_UseAnonymousFROM	Enable the use of "sip:anonymous@localhost" in FROM header of SIP INVITE when attempting to make an anonymous call	false
X_SessionRefresh	Enable session refresh signaling (with SIP Re-INVITE) during a connected call. This allows the OBi to detect if the connection with the peer is broken abnormally so it can release the call. Disable this option if the ITSP does not support Re-INVITE sent from the client device.	true
X_SessionTimer	Enable standard session timer behavior based on RFC4028	false
X_SessionExpires	Session Expires before value. If session refresh is enabled, OBi will refresh half-time before the session expires.	20
X_AccessList	A comma separated list of IP addresses such that the device only accepts SIP requests coming from one of the given addresses. If the list is empty, the device accepts SIP requests from any IP address	
X_InsertRTPStats	Enable the device to include a X-RTP-Stat header in a BYE request or 200 response to BYE request at the end of an established call. This header contains a summary of RTP statistics collected during the call.	true
X_MWISubscribe	Enable this option to have the device SUBSCRIBE to the message-summary event package to support MWI and VMWI service. Note that device handles NOTIFY of this event package regardless MWISubscribe is enabled or not	false
X_MWISubscribeURI	Blank implies to use the same URL as REGISTER for the TO and FROM header as well as the Request-URI Otherwise, if the URI does not contain '@', it is user as the userid field in TO/FROM header as well as the Request-URI, which are otherwise same as REGISTER If the URI contains '@', it is used in the TO and FROM header as well as the Request-URI as is Note that OBi device forms the Request-URI of SUBSCRIBE the same way as the TO header, with an additional port number	
X_MWISubscribeExpires	X_MWISubscribeExpires: periodic interval to renew SUBSCRIBE (default 3600s)	3600
X_RegSubscribe	Enable subscription to the reg event package	false
X_RegSubscribeExpires	reg event subscription expires value	3600
X_BLFSubscribeExpires	BLF subscription Expires value	3600
X_ShareLineMethod	The signaling method to use for shared line/share call appearances. The following choices are supported: - call-info - dialog;sla	call-info

	- dialog;ma	
X_CallInfoSubscribeExpires	Call-Info subscription Expires value	3600
X_BWCallParkSubscribeExpires	Call-Park subscription Expires value	3600
X_BWCallCenterSubscribeExpires	Call-Center subscription Expires value	3600
X_BWHotelingSubscribeExpires	hoteling subscription Expires value	3600
X_ASFeatureEventSubscribeExpires	as-event subscription Expires value	3600
X_LineSeizeSubscribeExpires	line-seize subscription Expires value	15
X_ProxyServerRedundancy	Enable proxy redundancy feature on the device. To use this feature, device registration must be enabled and the SIP Registration Server or Outbound Proxy Server must be configured as a domain name	false
X_SecondaryRegistration	Enable device to register with a secondary server in addition to the primary server. X_ProxyServerRedundancy must be enabled for this parameter to take effect	false
X_CheckPrimaryFallbackInterval	Interval in seconds at which the device should check the primary fallback list of candidate servers	60
X_CheckSecondaryFallbackInterval	Interval in seconds at which the device should check the secondary fallback list of candidate servers	60
X_ProxyFailoverResponseCodes	A list of failure response codes specified in the form of a digitmap string to trigger proxy failover. If only one digitmap is specified, it applies to REGISTER and INVITE requests. If two digitmaps are provided (separated by a comma), the first one applies to REGISTER and the second to INVITE.	([5-9]xx)
X_ProxyRequire	If this parameter is not blank, OBi will include a Proxy-Require header stating the value of this parameter in all SIP requests sent to the ITSP	
X_MaxForward	Value for the Max-Forward header in all SIP requests sent by the OBi	70
X_AcceptLanguage	If this parameter is not blank, OBi will include an Accept-Language header stating the value of this parameter in all SIP requests sent to the ITSP.	
X_DnsSrvAutoPrefix	Enable this option to let OBi automatically prepend a standard prefix to the domain name when querying DNS Server to resolve the ProxyServer or OutboundProxy name as a SRV record. The standard prefix is _sip._udp. for SIP over UDP, _sip._tcp. For SIP over TCP, and _sip._tls. for SIP over TLS.	false
X_Support100rel	Enable this option to turn on the support for RFC3262 (reliable provisional SIP responses). If enabled, OBi will announce this support in a SIP Supported header, and will require a caller to use this option if the caller also supports this feature.	false
X_UserEqPhone	Enable the insertion of user=phone parameter in INVITE Request-URI	false
X_UseTelURI	Enable the use of tel: in outbound SIP Request-URI and TO-URL	false
X_CallWaitingIndication	Choices are:	false

	No Alert-Info	
X_DiscoverPublicAddress	Enable this option to let the OBi use the public IP address and port it has discovered as its SIP Contact address	true
X_UsePublicAddressInVia		false
X_PublicIPAddress	A static public IPv4 address, if specified, will be used by the OBi to form its SIP Contact address	
X_UseRport	Enable this option to let the OBi insert a blank rport parameter in the VIA header our outbound SIP messages. This option should be turned off if you are using port forwarding on the external router to route inbound SIP messages to the OBi	true
X_UseCompactHeader	Enable the use of compact format SIP headers	false
X_FaxPassThroughSignal	Method to signal FAX passthrough to the peer. Available options are: ReINVITE RFC2833 Auto None If None is selected, FAX pass-through will not be signaled. If Auto is selected, RFC2833 is used if the peer has indicated support in SDP.	ReINVITE
X_IncludeMessageHash	Include a MD5 hash of all the SIP headers in an X-MD5-Hash header. A hash of the SDP is also included in an x-md5-hash SDP attribute	false
X_EchoServer	A server that can echo back SIP messages to the device	
X_EchoServerPort	The echo server listening port	5060
X_EnableRFC2543CallHold	Enable the device to recognize call hold signaling as used in RFC2543	false
Feature Configuration (VoiceService.1.VoiceProfile.k.SIP.), k = 1 – 6 for X = A – F, respectively		
X_CallParkMethod		Feature Code
X_AutoAnswerMethod	Method to signal to the called device to auto-answer the call. Choices are: - Call-Info: inserting ***answer-after=0*** parameter in a Call-Info header in the INVITE request - Alert-Info: inserting ***info=alert-autoanswer;delay=0*** parameter in an Alert-Info header in the INVITE request	Call-Info
X_DirectedCallPickupMethod		Feature Code
X_ShareLineMethod		call-info
X_BLFSubscribeExpires		3600
X_BWCallParkSubscribeExpires		3600
X_BW_CallCenterSubscribeExpires		3600
X_BWHotelingSubscribeExpires		3600
X_ASFeatureEventSbuscribeExpires		3600
X_LineSeizeSubscribeExpires		15
Feature Codes (VoiceService.1.VoiceProfile.k.X_FeatureCode.), k = 1 – 6 for X = A – F, respectively		
DirectedCallPickup		*98

CallPickup		*88
BargeIn		*33
Park		*68

ITSP Profile X – RTP Web Page (X = A, B, C, D, E, F)

Parameter	Description	Default Setting
RTP (VoiceService.1.VoiceProfile.k.RTP.), k = 1 – 6 for X = A – F, respectively		
LocalPortMin	Base of port range for tx/rx RTP with this SP	16600
LocalPortMax	Top of port range for tx/rx RTP with this SP	16798
KeepAliveInterval	Interval in seconds between sending keep alive packet on an RTP channel that is currently in idle (due to call hold for instance). RTP keepalive is disabled if the value of this parameter is set to 0.	0
DSCPMark	Diffserv code for outgoing RTP packets with this SP	0
X_UseSSL	Enable this option to force OBi to send RTP over a SSL channel when the ITSP is Google Voice	false
RTCP (VoiceService.1.VoiceProfile.2.RTP.RTCP.), k = 1 – 6 for X = A – F, respectively		
Enable	Enable RTCP operation	false
TxRepeatInterval	Interval in milliseconds between RTCP transmissions	10000
LocalCName	The local CNAME to use in RTCP message. By default OBi uses account-userid@local-ip-address as the local CNAME.	
X_RTCPMux	Enable RTCP-Mux operation (send and receive RTCP on the same local port as the corresponding RTP)	false
Jitter Buffer (VoiceService.1.VoiceProfile.k.RTP.JIB.), k = 1 – 6 for X = A – F, respectively		
Adaptive	Enable jitter buffer adaptation	true
MaximumSize	Maximum jitter buffer size in milliseconds	250
SetPoint	Initial play out delay in milliseconds	60
Target	Target play out delay in milliseconds	20
AdaptationSlope	Maximum adaptation slope in samples per 10ms	16

Voice Services

SP*n* Service Web Page (*n* = 1, 2, 3, 4, 5, 6)

The following configuration parameters are available on this page.

Parameter	**Description**	**Default Setting**
SPn Service (VoiceService.1.VoiceProfile.1.Line.n.), n = 1 – 6		
Enable	Enable this line	true
X_DisplayLabel		
X_DisplayNumber		
X_ServProvProfile	Select a Service Provider Profile for this service. Choices are A, or B	A
X_RingProfile	Select a Ring Profile to ring the PHONE port with for incoming	A

	calls on this service that are routed to the PHONE port. The ringing pattern will be taken from the given profile. Choices are A, or B	
X_CodecProfile	Select a Codec Profile for all calls on this service. Choices are A, or B	A
X_InboundCallRoute	Routing rule for directing incoming calls on this service. The default rule is to send all incoming calls to the PHONE port (ph). See *OBi Call Routing and Digit Map Section* for a description of the syntaxes for specifying this parameter	ph
X_RegisterEnable	Enable registration for this line. If set to YES, device sends periodic SIP REGISTER to the service provider according to the settings in the ITSP Profile. Otherwise, device does not send any SIP REGISTER for the service	true
X_NoRegNoCall	Enable this option to disallow incoming and outgoing calls if registration with the service provider is not successful	false
X_KeepAliveEnable	Enable sending keep alive message. If set to YES, device sends periodic keep-alive messages to the destination specified in X_KeepAliveServer and X_KeepAliveServerPort, at the interval specified in X_KeepAliveExpires. The content of this message is the ASCII string “keep-alive\r\n”	false
X_KeepAliveExpires	Keep alive period in seconds	15
X_KeepAliveServer	Hostname or IP address of keep alive server	
X_KeepAliveServerPort	UDP port of the keep alive server	5060
X_KeepAliveMsgType	The type of keep alive messages to send out periodically if keep-alive is enabled. It can be one of the following choices: - keep-alive: The string “keep-alive” - empty: A blank line - stun: A standard STUN binding request; device will use the binding response to form its contact address for REGISTRATION - custom: use the value of X_CustomeKeepAliveMsg (note: option not available on OBi100/OBi110)	keep-alive
X_CustomKeepAliveMsg	Defines the custom message to be used when X_KeepAliveMsgType is “custom”. The value should have the following format: `mtd=NOTIFY;event=<whatever>;user=<anyone>` Where - NOTIFY may be replaced by any other SIP method, such as PING, - event parameter is optional and is only applicable if method is NOTIFY. If event is not specified, the 'keep-alive' event will be used with NOTIFY - user parameter is optional; if not specieifed, the request-uri will not have a userid, and the TO header field will use the same userid as the FROM header (which is the local	

	account userid). If user is specified, it will be used as the userid in the Request-URI and TO header. SIP messages for keep-alive are sent only once without retransmission; response to the SIP messages are ignored by the OBi.	
X_UserAgentPort	UDP port where the device sends and listens for SIP messages	5060
X_UserAgentPorts	A comma separated list of (up to 10) alternative user agent ports to use when there is no response received from the SIP Registrar	
DirectoryNumber	Directory number associated with this service	
X_DefaultRing	Default ring pattern number to ring the PHONE port for incoming calls on this trunk that are routed to the PHONE port according to the InboundCallRoute of this service. The ring pattern is taken from the selected Ring Profile. Valid choices are 1-10	1
X_CallOnHoldRing	Pattern to ring PHONE port when holding a call on this trunk that has been connected to the PHONE port. Typically this is a very short distinctive ring pattern that serves as a reminder to the user that a call is being on hold. The ring pattern is taken from the selected Ring Profile. Valid choices are: NO Ring, or 1-10	8
X_RepeatDialRing	The ring pattern number to use to ring the PHONE port when a repeat dial operation on this trunk is successful as the called party is either ringing or answered	5
X_BargeInRing	Call Waiting Ring pattern to ring the PHONE port when the incoming call is requesting to barge-in. This is applicable in a call-waiting scenario on the PHONE port	4
X_CallParkedRing	Ring pattern to ring the PHONE port only as a reminder that there are some calls parked in the parking lot. This feature is applicable only in an OBiPLUS solution.	10
X_SipDebugOption	Enable sending of SIP signaling debug information to the syslog server (if one is configured on the device). Available choices are: Disable (do not send SIP signaling debug information) Log All Messages Log All Except REGISTER Messages	Disable
X_SipDebugExclusion	A list of SIP methods to exclude from the syslog for this SP service. For example: notify, subscribe	
X_SatelliteMode	Enable satellite mode on this trunk. In this mode, the user must explicitly sign on (using * code) to receive phone calls on this trunk. The SIP REGISTER sent by the OBi to the ITSP on this trunk will indicate if the user wants to sign on (and therefore takes over the incoming calls for this account). This feature is only applicable if the service is provided by an OBiPLUS system	false
X_AcceptResync	Control whether to accept a SIP NOTIFY request with event=resync to trigger a reboot of the device (so it can download new f/w or configuration upon boot up). Available	yes without authentication

	choices are: no (do not accept resync trigger) yes with authentication (accept after challenging the sender) yes without authentication (accept w/o challenging the sender)	
SIP Credentials (VoiceService.1.VoiceProfile.1.Line.*n*.SIP.), *n* = 1 – 6		
AuthUserName	The User ID to authenticate to a SIP UAS (User Agent Server) when an outbound SIP request sent by the device is challenged by the UAS with a 401 or 407 Response	
AuthPassword	The Password (corresponding to AuthUserName) to authenticate to a SIP UAS (User Agent Server) when an outbound SIP request sent by the device is challenged by the UAS with a 401 or 407 Response	
URI	This parameter affects the way the AOR is formed by the device in outbound SIP Requests. The AOR has the format: user@domain. If the value of URI is empty, device gets the user portion of its AOR from the AuthUserName, and the domain portion the value of ITSP Profile's UserAgentDomain if it is not empty, or that of the ProxyServer otherwise. If the value URI is not empty and does not contain "@", it is used as the user portion of the AOR while the domain portion is formed the usual way. If the value of URI contains "@', it is interpreted as a full AOR and device takes it as the AOR as is. Some Examples: 1) Let ProxyServer = *sip.myitsp.com*, AuthUserName = *4089991123*, URI=[empty], UserAgentDomain=[empty], then AOR = *4089991123@sip.myitsp.com* 2) Change UserAgentDomain to *users.myitsp.com*, then AOR = *4089991123@users.myitsp.com* 3) Change URI to bobdylan, then AOR = *bobdylan@users.myitsp.com* 4) Change URI to *bobdylan@superusers.myitsp.com*, then AOR = *bobdylan@superusers.myitsp.com* Note: In all cases, device uses AuthUserName and AuthUserPassword to compute authorization if challenged by a 401 or 407 response.	
X_MyExtension	An extension assigned to this line for inter-office calling and feature invocation	
X_ContactUserID	An alternative user-id to be used in Contact header. Enter **`Random`** to let the phone generate a random one.	
X_EnforceRequestUserID	Enforce incoming INVITE request user-id to match AuthUserName or X_ContactUserID	
X_ShareLine	Check this option if this is a shared line	false

X_ShareLineUserID	The (third-party) user-id to register for this line, if different from the account user-id.	
Calling Features (VoiceService.1.VoiceProfile.1.Line.*n*.CallingFeatures.), *n* = 1 – 6		
CallerIDName	Display name to identify the subscriber. The display name field is usually inserted in a FROM header in outbound SIP requests (such as INVITE) for the purpose of displaying a Caller ID Name on the recipient's device.	
MaxSessions	The maximum number of simultaneous calls that may be established on this service	2
CallForwardUnconditionalEnable	Enable call forwarding of all calls unconditionally by the device. If CallForwardUnconditionalNumber is blank, this parameter is treated as if it has been set to *No*. Note: It is possible for a user to set this parameter from the phone using a Star Code	false
CallForwardUnconditionalNumber	Directory number to forward all incoming calls on this service unconditionally. Maximum Length is 127 characters. Note: It is possible for a user to set this parameter from the phone using a Star Code	
CallForwardOnBusyEnable	Enable call forwarding of all incoming calls when the device is busy. If CallForwardOnBusyNumber is blank, this parameter is treated as if it has been set to *No*. Device is considered busy if one of the following conditions holds: This service already reaches the limit of simultaneous calls as specified in MaxSessions DND (Do Not Disturb) Service is enabled on this service If the call is routed to the PHONE port where the phone is in a busy state (such as ringing, dialing, playing reorder, or already having 2 calls in progress) Note: It is possible for a user to set this parameter from the phone using a Star Code	false
CallForwardOnBusyNumber	Directory number to forward all incoming calls on this service when the device is busy. Maximum Length is 127 characters. Note: It is possible for a user to set this parameter from the phone using a Star Code	
CallForwardOnNoAnswerEnable	Enable call forwarding of all incoming calls when the call is not answered after a period as specified in CallForwardOnNoAnswerRingCount. If CallForwardOnNoAnswerNumber is blank, this parameter is treated as if it has been set to No. Note: It is possible for a user to set this parameter from the phone using a Star Code	false
CallForwardOnNoAnswerNumber	Directory number to forward all incoming calls when the call is not answered after a period specified in CallForwardNoAnswerRingCount	

	Note: It is possible for a user to set this parameter from the phone using a Star Code	
CallForwardOnNoAnswerRingCount	Number of rings to be considered by the device as no answer to an incoming call. Note: 1 ring is approximately 6s	2
BlockedCallers	A comma separated list of up to 10 caller numbers to block from calling this service	
MWIEnableMask	Enable Message Waiting Indication from this service on one or more phone ports. It is specified as a bit mask, such that bit 0 for Phone 1, bit 1 for phone 2, etc.	255
X_VMWIEnableMask	Enable Visual Message Waiting Indication for this service on one or more phone ports. It is specified as a bit mask, such that bit 0 for Phone 1, bit 1 for phone 2, etc.	255
MessageWaiting	This is a state rather than a configuration parameter, that indicates if there are any new messages for this subscriber on the service provider's voicemail system	false
AnonymousCallBlockEnable	Enable blocking of Anonymous Calls on this service. Anonymous calls are rejected with a SIP 486 (Busy) response and Call Forward On Busy service is not applied. Note: It is possible for a user to set this parameter from the phone using a Star Code	false
AnonymousCallEnable	Enable masking of Caller-ID information for all outgoing calls. If enabled, the called party should perceive the call as coming from an anonymous caller. Note: It is possible for a user to set this parameter from the phone using a Star Code	false
DoNotDisturbEnable	Enable Do Not Disturb Service. If enabled, all incoming calls on this service are treated as if the device is busy. Note: It is possible for a user to set this parameter from the phone using a Star Code	false
X_BridgedOutboundCallMaxDuration	Limit on the call duration in seconds for all outbound calls that are bridged from the same or another trunk. A blank or 0 value implies the call duration is not limited.	
X_AcceptDialogSubscription	Enable the device to accept SUBSCRIBE to this trunk's dialog event package	false
X_SkipCallScreening	Enable the device to automatically skip call screening when the underlying ITSP is Google Voice	false
X_SMSNotify	Ring the phone on SMS reception from Google Voice and display the first few characters of the message as Caller-ID Note: Option available on OBi200/OBi202 only	false
X_XMPPPriority	XMPP Priority to assume by this client for Google Voice when there are multiple clients using the same account. Valid values are 0 (highest) or 32-127 Note: Option available on OBi200/OBi202 only	0

X_GTalkSimultaneousRing	Ring all other clients using the same Google Voice account at present. Note: Option available on OBi200/OBi202 only	true
X_SRTP	This is a drop down list with 3 choices: Disable SRTP = Do not use SRTP for all calls; the call will fail if the peer insists on using SRTP only Use SRTP Only = Require all calls to use SRTP; the call will fail if the peer does not support SRTP Use SRTP When Possible = Use SRTP for a call if the peer supports SRTP; otherwise fallback to use regular unencrypted SRTP	Disable SRTP
X_ASFeatureEventSubscribe	Enable subscription to the `as-feature-event` package	false
X_ShowDiversionHistory	Enable showing of diversion history of a call on the Calls screen	true
X_RTPReordURL	URL to upload RTP recording	
X_UrlDialingIPAddress	If enabled, dialing/calling an IP address will bypass the outbound proxy	true
X_UrlDialingDomainName	If enabled, dialing/calling a domain name will bypass the outbound proxy. For example, if you have a conference bridge number SP1(bridge@some-domin.com), you may need to set this option to false if you still need the call to this bridge to go through the outbound proxy	true
X_IncomingCallMustUseBoundCallKey	If enabled, the phone must assign an incoming to a Line Key with Function = Call Appearance and Service the same as the one the call is from. Otherwise, the phone may also assign the call to a Line Key with Function = Call Appearance and Service = $\{blank\}$ as a fall-back.	false
X_ConferenceBridge	The number of an external conference bridge to use with callson this service. Note: If the number also specifies the service to use, such as SP1(bridge@xyz-domain.com), the phone calls the number as is on the given service. Otherwise the phone applies it digit map and outbound call route setting to determine which service to use for the call	cbridge
Network Provided Services (VoiceService.1.VoiceProfile.1.Line.n.X_NetServices.), n = 1 – 6		
ACDAgent	Requires BroadSoft/Feature-Sync	false
AnonymousCall	Requires BroadSoft/Feature-Sync or BroadSoft/XSI	false
BroadWorksAnywhere	Requires Broadsoft/XSI	false
BuddyList	BroadSoft, Google Voice, or Generic XMPP Service	false
CallCenter	Requires BroadSoft	false
CallForwardAlways	Requires BroadSoft/Feature-Sync or BroadSoft/XSI	false
CallForwardBusy	Requires BroadSoft/Feature-Sync or BroadSoft/XSI	false
CallForwardNoAnswer	Requires BroadSoft/Feature-Sync or BroadSoft/XSI	false
CallLogs	Requires BroadSoft	false
CallPark	Park a call (soft key) against current extension with a Feature Access Code	false
CallParkStatus	Get notified if a call is parked against current extension. Requires BroadSoft	false
CallPickup	Retrieve a call (soft key) parked against current extension with a Feature Access Code	false

CallRecording	Requires BroadSoft	false
CallTrace	Requires BroadSoft	false
Directory	BroadSoft or similar directory	false
DispositionCode	Requires BroadSoft	false
DoNotDisturb	Requires BroadSoft/Feature-Sync or BroadSoft/XSI	false
Escalation	Requires BroadSoft	false
Executive	Requires BroadSoft	false
ExecutiveAssistant	Requires BroadSoft	false
Hoteling	Requires BroadSoft	false
RemoteOffice	Requires BroadSoft/XSI	false
SecurityClass	Requires BroadSoft/Feature-Sync	false
SimultaneousRing	Requires BroadSoft/XSI	false
ConferenceBridgeControl	View who is on the bridge and apply some control on selected participants (requires BroadSoft/XSI)	false
Netwrok Directory Setup (VoiceService.1.VoiceProfile.1.Line.*n*.X_NetServices.Dir.), *n* = 1 – 6		
URL	URL to download a Group directory. This parameter must be {*blank*} when usisng BroadWorks Directories where the URL to download the directory is formed internally based on BroadSoft XSI API.	
EnableSearch	Enable the Search soft key option on the directory pages.	true
ResultsPerPage	Number of results per page to suggest to the server. Default is 0 which means unspecified. Note that this is just a suggestion to the server which may return a different number of results per page; the phone always trys to display all the results returned by the server.	0
EnableGroup	Enable the Group Sub-Directory	true
EnableGroupCommon	Enable the GroupCommon Sub-Directory	true
EnableEnterprise	Enable the Enterprise Sub-Directory	true
EnableEnterpriseCommon	Enable the EnterpriseCommon Sub-Directory	true
EnablePersonal	Enable the Personal Sub-Directory	true
Buddy List Setup (VoiceService.1.VoiceProfile.1.Line.*n*.X_NetServices.BuddyList.), *n* = 1 – 6		
ServerType	Value can be `BroadSoft` or `Generic`	BroadSoft
Priority		33
VerifyServerCert	Boolean. Turn on/off verification of server certificate in a SSL connection with the XMPP server	false
QueryVcard	Boolean. Turn on/off reading of VCARD for each item in the roaster list. Turn on this option if you wish to get a picture of each buddy.	false
Buddy List Setup (VoiceService.1.VoiceProfile.1.Line.*n*.X_NetServices.ActionURL.), *n* = 1 – 6		
SyncURL	The Action URL the phone should execute every time when it starts up	
OBiStatusNotifyURL	The URL to send <obi-status> XML on start up and for status update. Default is SIP/NOTIFY. To use HTTP/POST, specifies http:// or https:// at the beginning of the URL	
OBiStatusFormat	Choose the format to report obi-status with. Value can be either `obi-status` or `dialog-info`. The former one is an obihai proprietary XML, the latter a <dialog-info> XML based on RFC4235	obi-status

OBiStatusFilter	Specifies which status changes to report to the given URL. The value is a string that is a comma-separated list of filter groups. There are three filter groups: • `Calls`: For status related to current calls on the phone, such as Call State, Caller-ID, etc. • `Services`: For status related to a voice service, such us up or down • `Features`: For status related to the setting of a feature, such as DND on/off Each filter group takes a comma-separated list of arguments to specify which status within each group to report. For the Calls group, the arguments can be one or more of the following: `sp1, sp2, sp3, sp4, sp5, sp6, pp, bt`. Each argument indicates the service whose call status are to be reported. For the Services group, the arguments can be one or more of the following: `sp1, sp2, sp3, sp4, sp5, sp6, pp, bt`. Each argument indicates the service whose statuses (except call status) are to be reported. For the Features group, each argument specifies a feature whose setting changes are to be reported; the following is a list of valid feature arguments: • `aans` (Auto Answer Intercom setting) • `aud` (Audio Path setting) • `bac` (Block Anonymous Call setting) • `bci` (Block Caller ID setting) • `cwa` (Call Waiting setting) • `dnd` (Do Not Disturb setting) • `dnr` (Do Not Ring setting) • `miss` (Number of new missed calls) • `mwi` (New Messages Waiting status) • `ncalls` (Number of calls) • `pg1` (Page Group 1 setting) • `pg2` (Page Group 2 setting) • `ring` (Ring status) Note: If OBiStatusFormat = dialog-info, only the Calls filter group will be used; the other filter groups are ignored For example: `Calls(sp1,bt),Services(sp1,sp2,bt),Features(dnd)`	Calls(sp1,sp2,sp3,sp4,sp5,sp6)

OBiTALK Service Configuration

Parameter	Description	Default Setting
OBiTALK Service Settings (VoiceService.1.X_P2P.1.)		
Enable	Enable the OBiTALK Service (the built-in free voice service that comes with every OBi Device)	true
LocalPort	The UDP or TCP port used by device to send and listens for OBiTALK messages	10000
TryMultiplePorts	Enable the unit to try a few random UDP ports until it can successfully join the OBiTALK network	true
DisplayName	Display name to identify the subscriber, for the purpose of	

	displaying a Caller ID Name on the recipient's device	
DigitMap	Digit map to restrict numbers that can be dialed or called with this service. See *OBi Call Routing and Digit Map Section* for a description of the syntaxes for specifying a Digit Map.	(<ob>xxxxxxxxx|obxxxxxxxxx)
InboundCallRoute	Routing rule for directing incoming calls on this service. The default rule is to send all incoming calls to the PHONE port (ph). See *OBi Call Routing and Digit Map Section* for a description of the syntaxes for specifying this parameter	ph,ph2
RingProfile	Select a Ring Profile to ring the PHONE port with when an incoming call is routed to the PHONE port. Choices are A, or B	A
CodecProfile	Select a Codec Profile to be used for all calls on this service. Choices are A, or B.	A
DefaultRing	Default ring pattern number to ring the PHONE port for incoming calls on this trunk that are routed to the PHONE port according to the InboundCallRoute of this service. The ring pattern is taken from the selected Ring Profile. Valid choices are 1-10	2
CallOnHoldRing	Pattern to ring PHONE port when holding a call on this trunk that has been connected to the PHONE port. Typically this is a very short distinctive ring pattern that serves as a reminder to the user that a call is being on hold. The ring pattern is taken from the selected Ring Profile. Valid choices are: NO Ring, or 1-10	8
RepeatDialRing	The ring pattern number to use to ring the PHONE port when a repeat dial operation on this trunk is successful as the called party is either ringing or answered	4
DTMFMethod	Method to pass DTMF digits to peer device. Available choices are: Inband - DTMF tone are sent as inband audio signal RFC2833 - DTMF tone events are relayed per RFC2833 SIPInfo - DTMF tones are relayed with SIP INFO request Auto - Method to use based on call setup negotiation (either Inband or RFC2833 may be negotiated)	Auto
UseFixedDurationRFC2833DTMF	When relaying DTMF digit events on this trunk using RFC2833, the RFC2833 RTP packets normally will keep streaming for as long as the digit is pressed. With this option set to TRUE, the device sends only one RTP digit event packet with a fixed duration of 150 ms regardless how long the digit has been pressed	false
Calling Features (VoiceService.1.X_P2P.1.CallingFeatures.)		
CallForwardUnconditionalEnable	Enable call forwarding of all calls unconditionally by the device. If CallForwardUnconditionalNumber is blank, this parameter is treated as if it has been set to *No*. Note: It is possible for a user to set this parameter from the phone using a Star Code	false
CallForwardUnconditionalNumber	Directory number to forward all incoming calls on this service unconditionally. Maximum Length is 127 characters. Note: It is possible for a user to set this parameter from the	

	phone using a Star Code	
CallForwardOnBusyEnable	Enable call forwarding of all incoming calls when the device is busy. If CallForwardOnBusyNumber is blank, this parameter is treated as if it has been set to *No*. Device is considered busy if one of the following conditions holds: This service already reaches the limit of simultaneous calls as specified in MaxSessions DND (Do Not Disturb) Service is enabled on this service If the call is routed to the PHONE port where the phone is in a busy state (such as ringing, dialing, playing reorder, or already having 2 calls in progress) Note: It is possible for a user to set this parameter from the phone using a Star Code	false
CallForwardOnBusyNumber	Directory number to forward all incoming calls on this service when the device is busy. Maximum Length is 127 characters. Note: It is possible for a user to set this parameter from the phone using a Star Code	
CallForwardOnNoAnswerEnable	Enable call forwarding of all incoming calls when the call is not answered after a period as specified in CallForwardOnNoAnswerRingCount. If CallForwardOnNoAnswerNumber is blank, this parameter is treated as if it has been set to No. Note: It is possible for a user to set this parameter from the phone using a Star Code	false
CallForwardOnNoAnswerNumber	Directory number to forward all incoming calls when the call is not answered after a period specified in CallForwardNoAnswerRingCount Note: It is possible for a user to set this parameter from the phone using a Star Code	
CallForwardOnNoAnswerRingCount	Number of rings to be considered by the device as no answer to an incoming call. Note: 1 ring is approximately 6s	2
BlockedCallers	A comma separated list of up to 10 caller numbers to block from calling this service	
MaxSessions	The maximum number of simultaneous calls that may be established on this service	2
AnonymousCallBlockEnable	Enable blocking of Anonymous Calls on this service. Anonymous calls are rejected with a SIP 486 (Busy) response and Call Forward On Busy service is not applied. Note: It is possible for a user to set this parameter from the phone using a Star Code	false
AnonymousCallEnable	Enable masking of Caller-ID information for all outgoing calls. If enabled, the called party should perceive the call as coming from an anonymous caller.	false

	Note: It is possible for a user to set this parameter from the phone using a Star Code	
DoNotDisturbEnable	Enable Do Not Disturb Service. If enabled, all incoming calls on this service are treated as if the device is busy. Note: It is possible for a user to set this parameter from the phone using a Star Code	false
Inbound Direct Dialing Authentication (VoiceService.1.X_P2P.1.VoiceGateway.)		
AuthMethod	The OBiTALK protocol allows incoming calls to indicate a target number that is different from this device's OBi number. The device in that case will attempt to establish and bridge the call to the target number according to the rules configured in the trunk's InboundCallRoute parameter. Hence this device acts as a gateway and the method is referred to direct dialing or 1-stage dialing (versus 2-stage dialing via the Auto-Attendant). Since the caller is not able to enter a PIN in such cases, an automated method based on signaling protocol must be used to authenticate the caller if authentication is required. OBi device offers the following choices for this purpose: None = Disable authentication HTTP Digest = Use HTTP Digest with User-ID and Password pairs. Note that at least one of AuthPassword*x* (*x*=1,2,3,4) must be specified, otherwise authentication is disabled.	HTTP Digest
AuthUserID1	One of 4 userids for authenticating direct dialing callers	
AuthPassword1	One of 4 passwords for authenticating direct dialing callers	
AuthUserID2	One of 4 userids for authenticating direct dialing callers	
AuthPassword2	One of 4 passwords for authenticating direct dialing callers	
AuthUserID3	One of 4 userids for authenticating direct dialing callers	
AuthPassword3	One of 4 passwords for authenticating direct dialing callers	
AuthUserID4	One of 4 userids for authenticating direct dialing callers	
AuthPassword4	One of 4 passwords for authenticating direct dialing callers	

Note: If AuthPassword is specified, AuthUserID may be set to blank to let the device use the default value which is a special hash of the AuthPassword. This is only applicable if the external gateway is also an OBi device that understands how to generate the default AuthUserID using the same hash function.

Auto Attendant Web Page

The following configuration parameters are available on this web page.

Parameter	Description	Default Setting
User Prompts (VoiceService.1.X_UserPrompt.)		
User<N>Description <N> = 1-10	A text string that describes the contents of this user prompt. You can click this parameter to invoke a page to upload an audio file for the prompt (.wav and .au files in 16-bit linear format at 8/11.025/16/22.05/32/44.1/48 kHz sample rate are supported).	
User<N>Length	This is a read-only status parameter. It shows the space occupied by this prompt in number of milliseconds	

<N> = 1-10		
SpacedUsed	This is a read-only status parameter. It shows the amount of recording space used in number of milliseconds	
SpaceAvailable	This is a read-only status parameter. It shows the amount of recording space remaining in number of milliseconds	
Auto Attendant (VoiceService.1.X_AA.1.)		
Enable	Enable AA. If enabled, the AA will answer an incoming call that has been routed to it after a period as specified in AnswerDelay. If disabled, the AA will not attempt to answer any incoming call.	true
DigitMap	Once the AA answers an incoming call, it presents the caller with an option to make a further call using one of the available voice services on the device. This Digit map serves to restrict the numbers that can be dialed or called via this AA option. See *OBi Call Routing and Digit Map Section* for a description of the syntaxes to specify a digit map.	([1-9]x?*(Mpli)| [1-9]|[1-9][0-9]| <00:$1>|[0-8]| **1(Msp1)| **2(Msp2)| **3(Msp3)| **4(Msp4)| **8(Mbt)| **9(Mpp)| (Mpli))
OutboundCallRoute	After the caller dials a number that is acceptable by the AA (according to its DigitMap) to make a further call, the device uses this outbound call routing rule to determine which service to make this call with. See *OBi Call Routing and Digit Map Section* for a description of the syntaxes to specify this parameter.	{([1-9]x?*(Mpli)):pp}, {0:ph}, {(<**1:>(Msp1)):sp1}, {(<**2:>(Msp2)):sp2}, {(<**3:>(Msp3)):sp3}, {(<**4:>(Msp4)):sp4}, {(<**8:>(Mbt)):bt}, {(<**9:>(Mpp)):pp}, {(Mpli):pli}
PrimaryLine	By primary line we mean the service that does not require any access code prefix (such as **1 or **9) when dialing; it is the default service to be used for making the call when no explicit access code prefix is entered. This parameter indicates to the device which voice service is considered as the primary line when dialing out via the Auto Attendant. Available choices are: SP1 Service (code = sp1) SP2 Service (code = sp2) SP3 Service (code = sp3) SP4 Service (code = sp4) SP5 Service (code = sp5) SP6 Service (code = sp6) OBiTALK Service (code = pp1) OBiBluetooth (code = bt1) Trunk Group 1 (code=tg1) Trunk Group 2 (code=tg2) The OBi device process the parameter by substituting of the	SP1 Service

	occurrences of *pli* and *(Mpli)* in DigitMap and OutboundCallRoute with the corresponding *code* and *(Mcode)*.	
AnswerDelay	Period of time in milliseconds that the AA will wait before answering an incoming call that has been routed to it	4000
NumberOnNoInput	In the case that the caller does not enter any option from the top level menu after the menu has been announced for 3 times, the AA directs the caller to the number specified in this parameter. If this number is not specified, the AA simply terminates the current call.	0 Note: According to the default DigitMap and OutboundCallRoute, calling 0 means ringing the Phone
UsePIN	Enable the use of PIN to authenticate callers when they select the option to make a further call. If PIN1, PIN2, PIN3, and PIN4 are all empty, device treats it as if UsePIN is set to No. Otherwise, the caller must enter one of the non-empty PIN in order to proceed,	false
PIN1	PIN code to make a call (must be all digits). Maximum Length = 15	
PIN2	PIN code to make a call (must be all digits). Maximum Length = 15	
PIN3	PIN code to make a call (must be all digits). Maximum Length = 15	
PIN4	PIN code to make a call (must be all digits). Maximum Length = 15	
Auto Attendant Prompts (VoiceService.1.X_AA.1.Prompt.)		
Welcome	Prompt List to replace the system's Welcome message. You may click this parameter to play it from the web browser	
InvalidPin	Prompt List to replace the system's InvalidPin message. You may click this parameter to play it from the web browser	
EnterPin	Prompt List to replace the system's EnterPin message. You may click this parameter to play it from the web browser	
MenuTitle	Prompt List to replace the system's MenuTitle message. You may click this parameter to play it from the web browser	
Menu	Prompt List to replace the system's Menu message. You may click this parameter to play it from the web browser	
PleaseWait	Prompt List to replace the system's PleaseWait message. You may click this parameter to play it from the web browser	
EnterNumber	Prompt List to replace the system's EnterNumber message. You may click this parameter to play it from the web browser	
Bye	Prompt List to replace the system's Bye message. You may click this parameter to play it from the web browser	

Gateways and Trunk Groups Web Page

Parameter	Description	Default Setting
Voice Gateway *n* (VoiceService.1.X_VoiceGateway.*n*.), *n* = 1 – 8		
Enable	Enable this voice gateway	true
Name	An arbitrary user-friendly name to identify this gateway (optional)	
AccessNumber	The gateway's OBiTALK number, including trunk information, such as: PP(ob200112334) or PP(ob300331456)	

	If the value is blank, device treats this VG as disabled. Starting with release 1.2, this can also be set to a SIP URL, such as: SP1(sip.mycompany.com:5060), or SP2(192.168.15.113)	
DigitMap	DigitMap for this VG. It can be referenced as (Mvg*n*)	(xx.)
AuthUserID	A User-ID to authenticate with the gateway	
AuthPassword	A Password to authenticate with the gateway	
Trunk Group *n* (VoiceService.1.X_TrunkGroup.*n*.), *n* = 1 – 4		
Enable	Enable this trunk group	true
Name	An arbitrary user friendly name to identify this trunk group (optional)	
TrunkList	A comma separated list of names of trunks to include in this trunk group.	For TG1, the default for OBi100 and OBi110 is: sp1,sp2 and for OBi202 is: sp1,sp2,sp3,sp4 For other TG, the default is (blank)
DigitMap	Digit map associated with this trunk group. It can be referenced as (Mtg*n*)	For TG1, the default is (1xxxxxxxxxx|<1>[2-9]xxxxxxxxx|011xx.|xx.) For other TG, the default is (xx.)

OBiBluetooth Web Page

The configuration parameters on this page are listed below.

Parameter	Description	Default Setting
OBiBluetooth (VoiceService.1.X_BT.1.)		
Enable	Enable this OBiBluetooth Service	true
DisplayLabel		
DisplayNumber		
AudioGateway		
DigitMap	Digit Map associated with this service. It can be referred as (Mbt*n*) in other digit maps and call routing rules	([2-9]xxxxxxS4|1xxxxxxxxxx|011xx.|[1-9]11S2|[1-9]xx)
InboundCallRoute	A set of call routing rules to control how to route incoming call on this service	ph
RingProfile	The Ring profile to use to ring a phone port for incoming calls on this service	A
DefaultRing	Default ring pattern to use to ring a phone port for incoming calls on this service	1
CallOnHoldRing	The ring pattern to use to ring a phone port to remind holding call on this service	8
DirectoryNumber	Informational only. The PSTN number associated with this service	
Calling Features (VoiceService.1.X_BT.1.CallingFeatures.)		
CallForwardUnconditionalEnable		false

CallForwardUnconditionalNumber		
CallForwardOnBusyEnable		false
CallForwardOnBusyNumber		
CallForwardOnNoAnswerEnable		false
CallForwardOnNoAnswerNumber		
CallForwardOnNoAnswerRingCount		2
BlockedCallers	A comma separated list of up to 10 caller numbers to block from calling this service	
AnonymousCallBlockEnable		false
DoNotDisturbEnable		false
BridgedOutboundCallMaxDuration		
AAAskForConfirm		
Device Settings (DeviceInfo.Bluetooth.Basic.)		
Discoverable	Read-Only. Indicate if the OBiBT on this channel is currently discoverable	false
ScanForDevices		false
PreferredPairedDevice	Select the preferred BT device when more than one paired devices are in range	None
PairedDevice*N* (*N* = 1 – 10)	Read-only. The name of the external paired BT device.	
RemovePairedDevice*N* (*N* = 1 – 10)	Check this box and press submit to remove this paired device	false

IP Phone Settings

Phone Settings Web Page

Parameter	Description	Default Setting
PhoneControl (VoiceService.1.Phone.Control.)		
ClearUserData	(Boolean)	false
ClearDownloadedDataCache	(Boolean)	false
Phone Settings (VoiceService.1.Phone.)		
DigitMap	This Digit map serves to restrict the numbers that can be dialed or called from the phone. See *OBi Call Routing and Digit Map Section* for a description of the syntaxes to specify a digit map.	([1-9]x?*(Mpli)|[1-9]S9| [1-9][0-9]S9|***|**0| **8(Mbt)|**1(Msp1)|**2(Msp2)| **3(Msp3)|**4(Msp4)| **9(Mpp)|(Mpli))
OutboundCallRoute	After the caller dials a number that is acceptable according to the DigitMap, OBi device uses this outbound call routing rule to determine which service to make this call with. See *OBi Call Routing and Digit Map Section* for a description of the syntaxes to specify this parameter	{([1-9]x?*(Mpli)):pp}, {**0:aa}, {***:aa2}, {(<**1:>(Msp1)):sp1}, {(<**2:>(Msp2)):sp2}, {(<**3:>(Msp3)):sp3}, {(<**4:>(Msp4)):sp4}, {(<**8:>(Mbt)):bt},

		{(<**9:>(Mpp)):pp}, {(Mpli):pli}
CallReturnDigitMaps	Call Return is the service where the user can call the last caller by dialing a star code (*69 by default). OBi device implements this service by remembering the number of the last caller in memory. However the stored information does not include any dialing prefix to tell the device which voice service to use to call back the last caller. This list of digit maps serve the purpose of mapping a caller's number to one that includes the desired dialing prefix used exclusively for call return service.	{pli:(xx.)}, {sp1:(<**1>xx.)}, {sp2:(<**2>xx.)}, {bt:(<**8>xx.)}, {pp:(<**9>xx.)}
PrimaryLine	By primary line we mean the service that does not require any access code prefix (such as **1 or **9) when dialing; it is the default service to be used for making the call when no explicit access code prefix is entered. This parameter indicates to the device which voice service is considered as the primary line when dialing out from the PHONE port. Available choices are: SP1 Service (code = sp1) SP2 Service (code = sp2) SP3 Service (code = sp3) SP4 Service (code = sp4) SP5 Service (code = sp5) SP6 Service (code = sp6) OBiTALK Service (code = pp1) OBiBluetooth (code=bt1) Trunk Group 1 (code=tg1) Trunk Group 2 (code=tg2) The OBi device process the parameter by substituting of the occurrences of *pli* and *(Mpli)* in DigitMap, OutboundCallRoute, and CallReturnDigitMaps with the corresponding code and (Mcode).	SP1 Service
ToneOnPrimaryServiceDown		
EnableCustomOBiPhoneXml		
EnableOBiPhoneXmlDownloadURL		
OBiPhoneXmlDownloadURL		
LastXmlDownloadURL		
LastXmlDownloadMD5		
GUI Menus (VoiceService.1.Phone.GUI.Menus.)		
MainMenu1		phone-book,calls,call-history,prod-info,preferences,settings,buddy,net srv,netdir

PreferencesMenu1		language;Language,timeFormat;Time Format,dateFormat;Date Format,hpTime;Auto Home Page,skin;Skin,bgpic;Background Picture,dring;Default Ringtone,dfont;Font,appCols;Home App Columns
PreferencesMenu2		packcalls;Pack Calls On Display,ssvr;Screen Saver,ssvrDelay;Screen Saver Delay,ssvrType;Screen Saver Type,ssvrLock;Require Passcode On Wake Up,ssvrPass;Wake Up Passcode,ssvrCustom;Screen Saver Show Custom Contents
PreferencesMenu3		brightness;Screen Brightness,HandsetVersion;Handset Version,audioDevice;Preferred Audio Device,headsetDevice;Preferred Headset Device,ehs;Electronic Hook Switch,dnd;Do Not Disturb,dnr;Do Not Ring
PreferencesMenu4		cfa.enable;Call Forward,cwa;Call Waiting,bac;Block Anonymous Call,bci;Anonymous Call,aans;Auto Answer Page,pg1;Join Page Group 1,pg2;Join Page Group 2,ringerVol;Ringer Volume
PreferencesMenu5		speakerVol;Speakerphone Volume,micGain;Speakerphone Mic Gain,handsetVol;Handset Volume,handsetGain;Handset Mic Gain,headsetRJ9Vol,RJ9 Headset Volume,headsetRJ9Gain;RJ9 Headset Mic Gain
PreferencesMenu6		headset35mmVol;3.5mm Headset Volume,headset35mmGain;3.5mm Headset Mic Gain,headsetBTVol;BT Headset Volume,headsetBTGain;BT Headset Mic Gain,EqEnable;Equalizer,AecEnable;AEC
SettingsMenu1		net;Network,wifi;WiFi,bt;Bluetooth,obiline;OBiLine,flash;Storage,ringfile;Ringtones,pkeys;Programmable Keys
SettingsMenu2		lkeys;Line Keys,sidecar1;Side Car 1,sidecar2;Side Car 2,voice;Voice Services,sd99,clrcache,admin;Device Admin,login
ProductInfoMenu1		model;Model,obinumber;OBi Number,mac;MAC Address,wfmac;WiFi MAC

		Address,serial;Serial Number
ProductInfoMenu2		swver;Software Version,hwver;Hardware Version,ztinfo;Customization Status,uptime;Up Time
NetServicesMenu1		acd,bci,bwanw,buddy,ccs,cfa,cfb,cfna,rec,dnd,dispcode,exec,xass,hotel,clog,dir,rmoff,secClass,simring
FactoryResetRequireAdminPassword	(Boolean) Require user to enter the Web Admin Password to factory reset the phone from the phone GUI	true
Calling Features (VoiceService.1.Phone.CallingFeatures.)		
MWIEnable	Enable MWI Signal (stutter dial tone) generation. If enabled, any SP voice service enabled on the device that has MWI Service enabled will trigger the generation of stutter dial tone if there are new voicemails for the subscriber on the service provider's voicemail system.	true
VMWIEnable	Enable VMWI Signal generation. If enabled, any SP voice service enabled on the device that has VMWI Service enabled will trigger the generation of VMWI signal if there are new voicemails for the subscriber on the service provider's voicemail system.	true
CallTransferEnable	Enable Call Transfer. Call Transfer, if enabled, is initiated by the user by hanging up the phone in one of the following scenarios: - One call on hold while a 2nd outgoing call ringing - One call on hold while a 2nd outgoing call connected - One call connected while a 2nd outgoing call ringing - 3-way conference with both calls connected If Call Transfer is disabled, hanging up the phone in the above scenarios simply ends all the calls, except for the one that is holding, which will remain on hold (cases 1 and 2).	true
ConferenceCallEnable	Enable 3-way Conference Call w/ local audio mixing. Conference Call, if enabled, is initiated by the user by hook flashing the phone in one of the following scenarios: - One call on hold while a 2nd outgoing call ringing - One call on hold while a 2nd outgoing call connected We refer to case (1) as an early conference, where the second conferee is still ringing; the other 2 parties may converse while hearing ringback tone in the background until the 3 party answers. In either case, the user can end the call with the second conferee by hook flashing another time and the call reverts to a 2-way call.	true

	If Conference Call service is disabled, then hook flashing the phone resumes the holding call but ends the second outgoing call in scenario (1), and swaps between the two calls in scenario (2) (as in a call waiting situation)	
UseExternalConferenceBridge	Use external conference bridge when starting a conference call	false
CallWaitingEnable	Enable call waiting service. Call Waiting is the situation where a new incoming call is routed to the PHONE port when there is already another call connected. If this service is enabled, OBi plays call-waiting tone to alert the user, as well as generates CWCID signal if CWCID is enabled. The user may then swap between the two calls by hook flashing. If the service is disabled, OBi rejects the incoming call as busy. Note: It is possible for the user to set this parameter from the phone using a Star Code	true
DoNotDisturbEnable		false
DoNotRingEnable		false
CallParkRingEnable	Enable playing of a short ring burst at configurable regular intervals for a configurable total duration wherever there is parked call in any of the call park orbit being monitored with a feature key (with the function Call Park Monitor)	false
AnonymousCallBlockEnable		false
AnonymousCallEnable		false
CallForwardUnconditionalEnable		false
CallForwardUnconditionalNumber		
CallForwardOnBusyEnable		false
CallForwardOnBusyNumber		
CallForwardOnNoAnswerEnable		false
CallForwardOnNoAnswerNumber		
CallForwardOnNoAnswerRingCount		2
JoinPageGroup1		false
JoinPageGroup2		false
AutoAnswerEnable		true
ToneProfile	Select a Tone Profile for call progress tone generation. Choices are A, or B	A
StarCodeProfile	Select a Star Code Profile for interpreting Star Codes entered by the user. Choices are None, A, or B. If value is set to None, no star code will be recognized by OBi device.	A
LastDialedNumber	Last number dialed out on the PHONE port	
LastCallerNumber	Last caller's number that rings the PHONE port	
AcceptMediaLoopback	Enable the device to accept incoming media loopback calls	true

MediaLoopbackAnswerDelay	Delay in milliseconds before the device answers an incoming media loopback call	0
MediaLoopbackMaxDuration	Maximum duration in seconds to allow for an inbound media loopback call. Set the value to blank or 0 to make it unlimited	0
RepeatDialInterval	Interval in seconds between redial in a repeat dial operation.	30
RepeatDialExpires	Duration of time in seconds when a repeat dial operation remains active.	1800
MOHServiceNumber	A number to call and bridge to get some audio played when the phone user holds the call	
PlaySITOnCallFailureCodes	Specify which INVITE failure response codes will trigger the OBi to play SIT tone to alert the phone user. It must be specified with a valid digitmap format. If nothing valid is specified, the OBi plays normal reorder tone when the outbound call fails.	(404|9xx)
InvalidNumberAnnouncementOption	Option to play an announcement of the error when user dials an invalid number Admissable Values: SIT_ONLY DEFAULT_ONLY SIT_AND_DEFAULT CUSTOM_ONLY SIT_AND_CUSTOM	SIT_ONLY
CustomInvalidNumberAnnouncement	The custom announcement to play when user dials an invalid number	
CallRemovalDelayOnPeerEndCall	Number of seconds to delay removal of the call from the screen when the far end ends the call. During that delay, reorder tone will be played if the call was connected (active) when it was ended	0
MergeConfereeCalls	A boolean option to enable of the grouping all the active conferee calls into a single call item and not showing each individual conferee call items on screen. The call state of this virtual call item is always **Conferencing**. Pressing the **End** soft key on this special call item ends all the conferee calls. Pressing the **Split** or **NewCall** soft key splits it up into individual conferee call items on screen while placing each conferee call on hold. Pressing **Join** soft key let the user drops out of the conference while transferring one conferee to the other so they can continue with the conversation.	false
TransferOnHangUpConference	A boolean option to enable transferring one conferee call to the other (in other words, Join the two conferees) when hanging up on a merged conference call. Otherwise, all the conferees are ended just like pressing the **End** soft key. Hanging up means either on-hook the handset while using the handset, turn off the speakerphone while using the speakerphone, or turn off the headset while using a headset.	false

Network Directory (VoiceService.1.Phone.NetDir.)		
Enable		
Name		
VoiceService		
Buddy List (VoiceService.1.Phone.BuddyList.)		
Enable		
Name		
MyPresence		
MyStatus		
VoiceService		
MyStatusHistory		
User Preferences Settings (VoiceService.1.Phone.SoftKeys.)		
CallFowardUnconditionalFeatureProvider		
DoNotDisturbFeatureProvider		
AnonymousCallFeatureProvider		
AnonymousCallBlockFeatureProvider		
MessageStatusFeatureProvider		
Timers (VoiceService.1.Phone.Timer.)		
HookFlashTimeMax		
HookFlashTimeMin		
ReorderDelayTime		
DigitMapLongTimer		
DigitMapShortTimer		
Page Group *n* (VoiceService.1.Phone.PageGroup.*n*.), *n* = 1, 2		
GroupName		
MulticastAddress		
MulticastPort		
TTL		
FullDuplex		
ParticipantName		
AudioCodec		
TxPacketSize		
RTCPTxInterval		
SilenceSuppression		
PushToTalk		

Line Keys

Parameter	Description	Default Setting
Line Key *n* (VoiceService.1.Phone.LineKey.*n*.), *n* = 1 – 24 (12 for OBi1032)		
Function		
Service		
Number		
Name		

Group		
Style		
MaxCalls		
Status		

Left Line Keys (OBi2000 series ony)

Parameter	Description	Default Setting
Left Line Key *n* (VoiceService.1.Phone.LeftLineKey.*n*.), *n* = 1 – 24 (for OBi2182; 12 for OBi2062/OBi2162)		
Function		
Service		
Number		
Name		
Group		
Style		
MaxCalls		
Status		

Programmable Keys

Note: In the OBi2000 series, the 9 programmable hard keys are configured with the same parameters as the corresponding 9 programmable keys.

Parameter	Description	Default Setting
Key *n* (VoiceService.1.Phone.ProgKey.*n*.), *n* = 1 – 8 (8 for OBi1032/1062, 1 for OBi1022, 9 for OBi2000 Series)		
Function		
Service		
Number		
Name		
Group		
Style		
Status		

Side Car *m*, *m* = 1, 2

Parameter	Description	Default Setting
Status (VoiceService.1.Phone.SideCar.*m*.)		
Connected	Side car connection status.	
Key *n* (VoiceService.1.Phone.SideCar.1.Key.*n*.), *n* = 1 – 16		
Function		
Service		
Number		
Name		
Group		

Style		
Status		

LED Settings

Parameter	Description	Default Setting
Call State (VoiceService.1.Phone.LED.Call.)		
Dialing		G
Trying		G
PeerRinging		G50,X50
Connected		G
Ringing		R50,X50
Holding		R500,X500
CallParked		R100,X1000
Error		G500,X500
ServiceDown		O
Service State (VoiceService.1.Phone.LED.Service.)		
Idle		G
InUse		G500,X500
Ringing		R50,X50
Holding		R500,X500
ServiceDown		O1000,G1000
SCA (VoiceService.1.Phone.LED.SCA.)		
Seized		R150,X150
Trying		R
PeerRinging		R
Connected		R
Held		G500,X500
PrivateHeld		R500,X500
Ringing		R50,X50
ServiceDown		O
BLF (VoiceService.1.Phone.LED.BLF.)		
Idle		X
CallParked		R100,X1000
Ringing		R50,X50
Busy		R
Holding		R500,X500
ServiceDown		O
Presence (VoiceService.1.Phone.LED.Presence.)		
Offline		X
Online		G
Busy		R
Away		G500,X500
ExtendedAway		R500,X500

ServiceDown		O
ACD Agent State (VoiceService.1.Phone.LED.ACD.)		
LoggedOff		X
Available		G
Unavailable		R
WrappingUp		R500,X500
ServiceDown		O
Feature State (VoiceService.1.Phone.LED.FeatureKey.)		
AnonymousCallEnabled		R
AnonymousCallDisabled		X
AnonymousCallServiceDown		O
AnonymousCallBlockEnabled		R
AnonymousCallBlockDisabled		X
AutoAnswerIntercomEnabled		X
AutoAnswerIntercomDisabled		R
CallForwardEnabled		R
CallForwardDisabled		X
CallForwardServiceDown		O
CallParkedYes		R
CallParkedNo		X
CallParkMonitorSErviceDown		O
CallWaitingEnabled		X
CallWaitingDisabled		R
DoNotDisturbEnabled		R
DoNotDisturbDisabled		X
DoNotDisturbServiceDown		O
DoNotRingEnabled		R
DoNotRingDisabled		X
ExecFilterEnabled		R
ExecFilterDisabled		X
ExecFilterServiceDown		O
ExecAssistEnabled		R
ExecAssistDisabled		X
ExecAssistDivertOn		R100,X500
ExecAssistServiceDown		O
HotelingGuestLoggedOn		R
HotelingGuestLoggedOff		X
HotelingServiceDown		O
NewMessagesWaitingYes		R
NewMessagesWaitingNo		X
MWIServiceDown		O
PageGroupJoined		R
PageGroupLeft		X
PageGroupMeTaling		R500,X500

PageGroupThemTalking		R50,X50
SecurityClassServiceDown		O
Feature State (VoiceService.1.Phone.LED.VMWI.)		
NewMessagesWaitingYes		R
NewMessagesWaitingNo		X
Disabled		X
Ringing		

Soft Key Sets

Parameter	Description	Default Setting
Soft Key Set (VoiceService.1.Phone.SoftKeySet.)		
Home		`redial,cfa,dnd,missed|lines`
SCAInUse		`sca.barge|,sca.monitor|,,newcall`
Dialtone		`redial,phbk,mode,lines`
Dialing		`redial,dial,mode,backspace`
OnDialing		`switch.line,dial,mode,backspace`
CallConnecting		`end,,,newcall`
CallConnected		`end,hold,conf,transfer,privhold,park,dispcode,escalate,trace,rec.start,rec.stop,rec.pause,rec.resume,tomobile`
CallHolding		`end,resume,add2conf,conf,transfer,park,dispcode,escalate,trace,rec.start,rec.stop,rec.pause,rec.resume,tomobile`
Ringing		`answer,reject,ignore`
CallParked		`pickup,,,newcall`
XferTrying		`end`
XferRinging		`end,,xfer.now`
XferConnected		`end,hold,resume,xfer.now`
ConfTrying		`end`
ConfRinging		`end,,conf.now`
ConfConnected		`end,hold,resume,conf.now`
NetServices		`ns.bci,ns.dnd,ns.cfa,ns.acd`

Feature Key Customization

Parameter	Description	Default Setting
Call Appearance (VoiceService.1.Phone.CstmLineKey.Call.)		
Enable	Boolean. Enable the use of customized feature key display format for key function Call Appearance	`false`

TextLine1	Use this text for line 1 when the key has no calls	`@text1`
TextLine2	Use this text for line 2 when the key has no calls	`@text2`
TextLine1InCall	Use this text for line 1 when the key has one or more calls	`@text1`
TextLine2InCall	Use this text for line 2 when the key has one or more calls	`@text2`
Busy Lamp Field (VoiceService.1.Phone.CstmLineKey.BLF.)		
Enable	Boolean. Enable the use of customized feature key display format for key function Busy Lamp Field	`false`
TextLine1	Use this text for line 1 when the monitored extension has no calls	`@text1`
TextLine2	Use this text for line 2 when the monitored extension has no calls	`@text2`
TextLine1InCall	Use this text for line 1 when the monitored extension has one or more calls	`@text1`
TextLine2InCall	Use this text for line 2 when the monitored extension has one or more calls	`@text2`
Speed Dial (VoiceService.1.Phone.CstmLineKey.SpeedDial.)		
Enable	Boolean. Enable the use of customized feature key display format for key function Speed Dial	`false`
TextLine1		`@text1`
TextLine2		`@text2`
Action URL (VoiceService.1.Phone.CstmLineKey.ActionURL.)		
Enable	Boolean. Enable the use of customized feature key display format for key function Action URL	`false`
TextLine1		`@text1`
TextLine2		`@text2`
Group n (VoiceService.1.Phone.CstmLineKey.Group.n.), n = 1, 2, 3, 4		
Enable	Boolean. Enable the use of customized feature key display format for key function Action URL	`false`
TextLine1		`@text1`
TextLine2		`@text2`
Icon		`@icon`

Screen Item Customization

Parameter	Description	Default Setting
Screen Item (VoiceService.1.Phone.Screen.)		
RingItem	The value is a <ScreenItem> XML to customize the format of the Ringing Calls display on the phone screen	
CallItem	The value is a <ScreenItem> XML to customize the format the Current Calls display on the phone screen	
CallTransferItem	The value is a <ScreenItem> XML to	

	customize the format the Call Transfer display on the phone screen. Unlike the last two items which are items in a table, there is only 1 CallTransferItem	

ScreenSaverCustomContents	The value is a <ScreenItem> XML to customize the contents to be shown on top of the current screen saver picture when the screen saver is being shown.	For OBi1032/1062 and OBi2000 Series
		`<ScreenItem>`
		`  <time align='center' ypos='60'`
		`  font='@gfont-bold' size='60'`
		`  format='H-m-s APM'`
		`  shadow='2,1,0x444444'/>`
		`  <date align='center' ypos='134'`
		`  font='@gfont' size='30'`
		`  format='w     m d, y'`
		`  shadow='2,1,0x444444'/>`
		`</ScreenItem>`
		
		For OBi1022
		`<ScreenItem>`
		`  <time align='center' ypos='44'`
		`  font='@gfont-bold' size='60'`
		`  format='@timeFmt'`
		`  shadow='2,1,0x444444'/>`
		`  <date align='center' ypos='118'`
		`  font='@gfont' size='30'`
		`  format='w     mn/d/y'`
		`  shadow='2,1,0x444444'/>`
		`</ScreenItem>`

SoftKeyStyles	Contains a list of up to 32 <style> elements to be used for format a soft key. The enclosing root element must be <SoftKeyStyles>. To use a style, specify a style attribute in the custom soft key (in a soft key set parameter), with the value equal to id attribute of the corresponding <style> element. If no style attribute is specified or the given style is not found, the <style> with id="default" is used for the soft key, if one is found. Otherwise the internally defined style is used for the soft key.	`<SoftKeyStyles`
		`  bgimg="http://abc.com/sk1.png"`
		`  hlimg="http://abc.com/sk2.png">`
		`  <style id="default">`
		`  <label align="center"/>`
		`  <icon align="center"/>`
		`  </style>`
		`  <style id= "left">`
		`  <label align="left"/>`
		`  <icon align="right"/>`
		`  </style>`
		`  <style id= "right">`
		`  <label align="right"/>`
		`  <icon align="left"/>`
		`  </style>`
		`</SoftKeyStyles>`

Codec Profiles

Codec Profile *X* Web Page (X = A, B)

Parameter	Description	Default Setting
G711U Codec (VoiceService.1.VoiceProfile.1.Line.*n*.Codec.List.1.), n = 1, 2 for X = A, B respectively		
Codec	Codec Name	G711U
BitRate	Bit rate in bits/sec. Note: Informational only; not configurable	64000
Enable	Enable this codec	true
SilenceSuppression	Enable silence suppression for this codec	false
PacketizationPeriod	Packet size in ms	20
Priority	Priority assigned to this codec (1 is the highest)	3
PayloadType	Standard payload type for this codec Note: Informational only; not configurable	0

G711A Codec (VoiceService.1.VoiceProfile.1.Line.*n*.Codec.List.2.) , *n* = 1, 2 for *X* = A, B respectively		
Codec	Codec Name	G711A
BitRate	Bit rate in bits/sec Note: Informational only; not configurable	64000
Enable	Enable this codec	true
SilenceSuppression	Enable silence suppression for this codec	false
PacketizationPeriod	Packet size in ms	20
Priority	Priority assigned to this codec (1 is the highest)	4
PayloadType	Standard payload type for G711-alaw Note: Informational only; not configurable	8
G729 Codec (VoiceService.1.VoiceProfile.1.Line.*n*.Codec.List.3.) , *n* = 1, 2 for *X* = A, B respectively		
Codec	Codec Name	G729
BitRate	Bit rate in bits/sec Note: Informational only; not configurable	8000
Enable	Enable this codec	true
SilenceSuppression	Enable silence suppression for this codec	false
PacketizationPeriod	Packet size in ms	20
Priority	Priority assigned to this codec (1 is the highest)	5
PayloadType	Standard payload type for G.729 Note: Informational only; not configurable	18
G726R32 Codec (VoiceService.1.VoiceProfile.1.Line.*n*.Codec.List.4.) , *n* = 1, 2 for *X* = A, B respectively		
Codec	Codec Name	G726-32
BitRate	Bit rate in bits/sec Note: Informational only; not configurable	32000
Enable	Enable this codec	true
SilenceSuppression	Enable silence suppression for this codec	false
PacketizationPeriod	Packet size in ms	20
Priority	Priority assigned to this codec (1 is the highest)	6
PayloadType	Dynamic Payload type for this codec. Valid range is 96-127	104
G726R16 Codec (VoiceService.1.VoiceProfile.1.Line.*n*.Codec.List.5.) , *n* = 1, 2 for *X* = A, B respectively		
Codec	Codec Name	G726-16
BitRate	Bit rate in bits/sec Note: Informational only; not configurable	16000
Enable	Enable this codec	true
SilenceSuppression	Enable silence suppression for this codec	false
PacketizationPeriod	Packet size in ms	20
Priority	Priority assigned to this codec (1 is the highest)	7
PayloadType	Dynamic Payload type for this codec. Valid range is 96-127	102
G726R24 Codec ((VoiceService.1.VoiceProfile.1.Line.*n*.Codec.List.6.) , *n* = 1, 2 for *X* = A, B respectively		
Codec	Codec Name	G726-24
BitRate	Bit rate in bits/sec Note: Informational only; not configurable	24000
Enable	Enable this codec	true
SilenceSuppression	Enable silence suppression for this codec	false

PacketizationPeriod	Packet size in ms	20
Priority	Priority assigned to this codec (1 is the highest)	8
PayloadType	Dynamic Payload type for this codec. Valid range is 96-127	103
G726R40 Codec (VoiceService.1.VoiceProfile.1.Line.n.Codec.List.7.) , n = 1, 2 for X = A, B respectively		
Codec	Codec Name	G726-40
BitRate	Bit rate in bits/sec Note: Informational only; not configurable	40000
Enable	Enable this codec	true
SilenceSuppression	Enable silence suppression for this codec	false
PacketizationPeriod	Packet size in ms	20
Priority	Priority assigned to this codec (1 is the highest)	9
PayloadType	Dynamic Payload type for this codec. Valid range is 96-127	105
iLBC Codec (VoiceService.1.VoiceProfile.1.Line.n.Codec.List.8.) , n = 1, 2 for X = A, B respectively		
Codec	Codec Name	G726-40
BitRate	Bit rate in bits/sec Two values to choose from: 13333 bps or 15200 bps	40000
Enable	Enable this codec	true
SilenceSuppression	Enable silence suppression for this codec	false
PacketizationPeriod	Packet size in ms. Must be multiples of 30 for 13333 bps or multiples of 20 for 15200 bps	30
Priority	Priority assigned to this codec (1 is the highest)	10
PayloadType	Dynamic Payload type for this codec. Valid range is 96-127	98
G722 Codec (VoiceService.1.VoiceProfile.1.Line.n.Codec.List.9.) , n = 1, 2 for X = A, B respectively		
Codec	Codec Name for this RTP event, as used in SDP	G722
BitRate	64000	64000
Enable	Enable this codec	true
SilenceSuppression	Enable silence suppression for this codec	false
PacketizationPeriod	Packet size in milliseconds	20
Priority	Priority assigned to this codec (1 is the highest)	2
PayloadType		9
OPUS Codec (VoiceService.1.VoiceProfile.1.Line.n.Codec.List.10.) , n = 1, 2 for X = A, B respectively		
Codec	Codec Name for this RTP event, as used in SDP. Note that some server may expect this to be lower case `opus`.	OPUS
BitRate	Variable	20000
Enable	Enable this codec	true
SilenceSuppression	Enable silence suppression for this codec	false
PacketizationPeriod	Packet size in milliseconds	20
Priority	Priority assigned to this codec (1 is the highest)	1
PayloadType		109
Telephone Event (VoiceService.1.VoiceProfile.1.Line.n.Codec.X_TelephoneEvent.) , n = 1, 2 for X = A, B respectively		
Codec	Codec Name for this RTP event, as used in SDP	telephone-event
Enable	Enable this codec	true
PayloadType	Dynamic Payload type to be used for RFC2833 telephone (DTMF) events. Valid range is 96-127	101

Encap RTP (VoiceService.1.VoiceProfile.1.Line.*n*.Codec.X_EncapRTP.) , *n* = 1, 2 for *X* = A, B respectively		
Codec	Codec Name. This codec is used to encapsulate RTP packets during a packet loopback call	encaprtp
PayloadType	Dynamic Payload type for this codec. Valid range is 96-127	107
Loopback Primer (VoiceService.1.VoiceProfile.1.Line.*n*.Codec.X_LoopbackPrimer.) , *n* = 1, 2 for *T X* = A, B respectively		
Codec	Codec Name. The codec is used by the OBi when acts as a media loopback mirror and before receiving any packets from the loopback source during a media loopback call	loopbkprimer
PayloadType	Dynamic Payload type for this codec. Valid range is 96-127	108
Codec Settings (VoiceService.1.VoiceProfile.1.Line.*n*.Codec.X_Settings.) , *n* = 1, 2 for *X* = A, B respectively		
G726BitPacking	Two values to choose from: big-endian or little-endian	big-endian

Tone Profiles

Tone Profile *X* Web Page (*X* = A, B)

Parameter	Description	Default Setting
Dial Tone (VoiceService.1.VoiceProfile.*n*.Tone.Description.1.) , *n* = 1, 2 for *X* = A, B respectively		
ToneName	Dial Tone	
TonePattern	Obihai Tone Pattern Script	350-18,440-18;20
Ringback Tone (VoiceService.1.VoiceProfile.*n*.Tone.Description.2.) , *n* = 1, 2 for *X* = A, B respectively		
ToneName	Ringback Tone	
TonePattern	Obihai Tone Pattern Script	440-18,480-18;-1;(2+4)
Busy Tone (VoiceService.1.VoiceProfile.*n*.Tone.Description.3.) , *n* = 1, 2 for *X* = A, B respectively		
ToneName	Busy Tone	
TonePattern	Obihai Tone Pattern Script	480-18,620-18;10;(.5+.5)
Reorder Tone (VoiceService.1.VoiceProfile.*n*.Tone.Description.4.) , *n* = 1, 2 for *X* = A, B respectively		
ToneName	Reorder tone or Fastbusy	
TonePattern	Obihai Tone Pattern Script	480-18,620-18;10;(.25+.25)
Confirmation Tone (VoiceService.1.VoiceProfile.*n*.Tone.Description.5.) , *n* = 1, 2 for *X* = A, B respectively		
ToneName	Confirmation Tone	
TonePattern	Obihai Tone Pattern Script	600-18;1;(.2+.2)
Holding Tone (VoiceService.1.VoiceProfile.*n*.Tone.Description.6.) , *n* = 1, 2 for *X* = A, B respectively		
ToneName	Holding Tone played when peer holding the call	
TonePattern	Obihai Tone Pattern Script	800-18;30;(.1+10)
Second Dial Tone (VoiceService.1.VoiceProfile.*n*.Tone.Description.7.) , *n* = 1, 2 for *X* = A, B respectively		
ToneName	Second Dial Tone played when dialing second call in a 3-way call	
TonePattern	Obihai Tone Pattern Script	385-18,484-18;20
Stutter Dial Tone (VoiceService.1.VoiceProfile.*n*.Tone.Description.8.) , *n* = 1, 2 for *X* = A, B respectively		
ToneName	Stutter Dial Tone	
TonePattern	Obihai Tone Pattern Script	350-18,440-18;20;2(.1+.1);()
Howling Tone (VoiceService.1.VoiceProfile.*n*.Tone.Description.9.) , *n* = 1, 2 for *X* = A, B respectively		

ToneName	Howling Tone for off-hook warning	
TonePattern	Obihai Tone Pattern Script	480+3,620+3;10;(.125+.125)
Prompt Tone (VoiceService.1.VoiceProfile.*n*.Tone.Description.10.) , *n* = 1, 2 for *X* = A, B respectively		
ToneName	Prompt Tone to prompt user to enter a number for configuration, such as speed dial	
TonePattern	Obihai Tone Pattern Script	480-16;20
Call Forward Dial Tone (VoiceService.1.VoiceProfile.*n*.Tone.Description.11.) , *n* = 1, 2 for *X* = A, B respectively		
ToneName	Call Forward Dial Tone (Special dial tone to indicate call-forward-all active)	
TonePattern	Obihai Tone Pattern Script	350-18,440-18;20;(.2+.2)
DND Dial Tone (VoiceService.1.VoiceProfile.*n*.Tone.Description.20.) , *n* = 1, 2 for *X* = A, B respectively		
ToneName	DND Dial Tone (Special dial tone to indicate DND active)	
TonePattern	Obihai Tone Pattern Script	350-18,440-18;20;(.2+.2)
Conference Tone (VoiceService.1.VoiceProfile.*n*.Tone.Description.12.) , *n* = 1, 2 for *X* = A, B respectively		
ToneName	Conference Tone (Indicates conference has started)	
TonePattern	Obihai Tone Pattern Script	350-16;10;(.1+.1,.1+9.7)
SIT Tone 1 (VoiceService.1.VoiceProfile.*n*.Tone.Description.13.) , *n* = 1, 2 for *X* = A, B respectively		
ToneName	Special Information Tone - 1	
TonePattern	Obihai Tone Pattern Script	985-16,1428-16,1777-16;20;(1/.380+0,2/.380+0,4/.380+0,0/0+4)
SIT Tone 2 (VoiceService.1.VoiceProfile.*n*.Tone.Description.14.) , *n* = 1, 2 for *X* = A, B respectively		
ToneName	Special Information Tone - 2	
TonePattern	Obihai Tone Pattern Script	914-16,1371-16,1777-16;20;(1/.274+0,2/.274+0,4/.380+0,0/0+4)
SIT Tone 3 (VoiceService.1.VoiceProfile.*n*.Tone.Description.15.) , *n* = 1, 2 for *X* = A, B respectively		
ToneName	Special Information Tone - 3	
TonePattern	Obihai Tone Pattern Script	914-16,1371-16,1777-16;20;(1/.380+0,2/.380+0,4/.380+0,0/0+4)
SIT Tone 4 (VoiceService.1.VoiceProfile.*n*.Tone.Description.16.) , *n* = 1, 2 for *X* = A, B respectively		
ToneName	Special Information Tone - 4	
TonePattern	Obihai Tone Pattern Script	985-16,1371-16,1777-16;20;(1/.380+0,2/.380+0,4/.380+0,0/0+4)
Outside Dial Tone (VoiceService.1.VoiceProfile.*n*.Tone.Description.17.) , *n* = 1, 2 for *X* = A, B respectively		
ToneName	Outside Dial Tone	
TonePattern	Obihai Tone Pattern Script	385-16;10
Paging Tone (VoiceService.1.VoiceProfile.*n*.Tone.Description.19.) , *n* = 1, 2 for *X* = A, B respectively		
ToneName	Paging Tone (The short tone played before playing the caller's voice)	
TonePattern	Obihai Tone Pattern Script	480-16;1;(.2+.2)

Ring Profiles

Ring Profile X Web Page (X = A, B)

Parameter	Description	Default Setting
Call Waiting Tone 1 (VoiceService.1.VoiceProfile.n.Tone.Description.21.), n = 1,2 for X=A,B respectively		
ToneName	Distinctive Call Waiting Tone 1	Bellcore-dr1
TonePattern	Obihai Tone Pattern Script	440-18;30;(.25+10)
Call Waiting Tone 2 (VoiceService.1.VoiceProfile.n.Tone.Description.21.), n = 1,2 for X=A,B respectively		
ToneName	Distinctive Call Waiting Tone 2	Bellcore-dr2
TonePattern	Obihai Tone Pattern Script	440-18;30;(.1+.1,.3+.1,.1+10)
Call Waiting Tone 3 (VoiceService.1.VoiceProfile.n.Tone.Description.21.), n = 1,2 for X=A,B respectively		
ToneName	Distinctive Call Waiting Tone 3	Bellcore-dr3
TonePattern	Obihai Tone Pattern Script	440-18;30;(.1+.1,.1+10)
Call Waiting Tone 4 (VoiceService.1.VoiceProfile.n.Tone.Description.21.), n = 1,2 for X=A,B respectively		
ToneName	Distinctive Call Waiting Tone 4	Bellcore-dr4
TonePattern	Obihai Tone Pattern Script	440-18;30;(.1+.1,.1+.1,.1+10)
Call Waiting Tone 5 (VoiceService.1.VoiceProfile.n.Tone.Description.21.), n = 1,2 for X=A,B respectively		
ToneName	Distinctive Call Waiting Tone 5	Bellcore-dr5
TonePattern	Obihai Tone Pattern Script	440-18;30;(.3+.1,.1+.1,.3+10)
Call Waiting Tone 6 (VoiceService.1.VoiceProfile.n.Tone.Description.21.), n = 1,2 for X=A,B respectively		
ToneName	Distinctive Call Waiting Tone 6	User-dr1
TonePattern	Obihai Tone Pattern Script	440-18;30;(.1+.1,.3+.2,.3+10)
Call Waiting Tone 7 (VoiceService.1.VoiceProfile.n.Tone.Description.21.), n = 1,2 for X=A,B respectively		
ToneName	Distinctive Call Waiting Tone 7	User-dr2
TonePattern	Obihai Tone Pattern Script	440-18;30;(.3+.1,.3+.1,.1+10)
Call Waiting Tone 8 (VoiceService.1.VoiceProfile.n.Tone.Description.21.), n = 1,2 for X=A,B respectively		
ToneName	Distinctive Call Waiting Tone 8	User-dr3
TonePattern	Obihai Tone Pattern Script	440-18;30;(.3+2)
Call Waiting Tone 9 (VoiceService.1.VoiceProfile.n.Tone.Description.21.), n = 1,2 for X=A,B respectively		
ToneName	Distinctive Call Waiting Tone9	User-dr4
TonePattern	Obihai Tone Pattern Script	440-18;30;(.3+2)
Call Waiting Tone 10 (VoiceService.1.VoiceProfile.n.Tone.Description.21.), n = 1,2 for X=A,B respectively		
ToneName	Distinctive Call Waiting Tone 10	User-dr5
TonePattern	Obihai Tone Pattern Script	440-18;30;(.3+2)
Ring Pattern 1 (VoiceService.1.VoiceProfile.1.Line.n.Ringer.Description.1.), n = 1,2 for X=A,B respectively		
RingName		Bellcore-dr1
RingPattern		60;(2+4)
Ring Pattern 2 (VoiceService.1.VoiceProfile.1.Line.n.Ringer.Description.2.), n = 1,2 for X=A,B respectively		
RingName		Bellcore-dr2
RingPattern		60;(.3+.2,1+.2,.3+4)
Ring Pattern 3 (VoiceService.1.VoiceProfile.1.Line.n.Ringer.Description.3.), n = 1,2 for X=A,B respectively		
RingName		Bellcore-dr3

RingPattern		60;(.8+.4,.8+4)
Ring Pattern 4 (VoiceService.1.VoiceProfile.1.Line.*n*.Ringer.Description.4.), *n* = 1,2 for *X*=A,B respectively		
RingName		Bellcore-dr4
RingPattern		60;(.4+.2,.3+.2,.8+4)
Ring Pattern 5 (VoiceService.1.VoiceProfile.1.Line.*n*.Ringer.Description.5.), *n* = 1,2 for *X*=A,B respectively		
RingName		Bellcore-dr5
RingPattern		60;(.2+.2,.2+.2,.2+.2,1+4)
Ring Pattern 6 (VoiceService.1.VoiceProfile.1.Line.*n*.Ringer.Description.6.), *n* = 1,2 for *X*=A,B respectively		
RingName		User-dr1
RingPattern		60;(.2+.4,.2+.4,.2+4)
Ring Pattern 7 (VoiceService.1.VoiceProfile.1.Line.*n*.Ringer.Description.7.), *n* = 1,2 for *X*=A,B respectively		
RingName		User-dr2
RingPattern		60;(.4+.2,.4+.2,.4+4)
Ring Pattern 8 (VoiceService.1.VoiceProfile.1.Line.*n*.Ringer.Description.8.), *n* = 1,2 for *X*=A,B respectively		
RingName		User-dr3
RingPattern		60;(.25+9.75)
Ring Pattern 9 (VoiceService.1.VoiceProfile.1.Line.*n*.Ringer.Description.9.), *n* = 1,2 for *X*=A,B respectively		
RingName		User-dr4
RingPattern		60;(.25+9.75)
Ring Pattern 10 (VoiceService.1.VoiceProfile.1.Line.*n*.Ringer.Description.10.), *n* = 1,2 for *X*=A,B respectively		
RingName		User-dr5
RingPattern		60;(.25+9.75)
Ring Pattern 11 (VoiceService.1.VoiceProfile.1.Line.*n*.Ringer.Description.11.), *n* = 1,2 for *X*=A,B respectively		
RingName		User-dr5
RingPattern		60;(.25+9.75)

Star Code Profiles

Star Code Profile *X* Web Page (*X* = A, B)

Parameter	Description	Default Setting
Code1	Default = Redial Star Code	*07, Redial, call($Ldn)
Code2	Default = Call Return Star Code	*69, Call Return, call($Lcn)
Code3	Default = Block Caller ID (Persistent) Star Code	*81, Block Caller ID, set($Bci,1)
Code4	Default = Unblock Caller ID (Persistent) Star Code	*82, Unblock Caller ID, set($Bci,0)
Code5	Default = Block Caller ID Once Star Code	*67, Block Caller ID Once, set($Bci1,1)
Code6	Default = Unblock Caller ID Once Star Code	*68, Unblock Caller ID Once, set($Ubci1,1)
Code7	Default = Call Forward Unconditional Star Code	*72, Cfwd All, coll($Cfan), set($Cfa,1)
Code8	Default = Disable Call Forward Unconditional Star Code	*73, Disable Cfwd All, set($Cfa, 0)
Code9	Default = Call Forward on Busy Star Code	*60, Cfwd Busy, coll($Cfbn), set($Cfb,1)
Code10	Default = Disable Call Forward on Busy Star Code	*61, Disable Cfwd Busy, set($Cfb, 0)
Code11	Default = Call Forward on No Answer Star Code	*62, Cfwd No Ans, coll($Cfnn), set($Cfn,1)
Code12	Default = Disable Call Forward on No Answer Star Code	*63, Disable Cfwd No Ans, set($Cfn,0)
Code13	Default = Block Anonymous Calls Star Code	*77, Block Anonymous Call, set($Bac,1)

Code14	Default = Unblock Anonymous Calls Star Code	*87, Unblock Anonymous Call, set($Bac,0)
Code15	Default = Enable Call Waiting Star Code	*56, Enable Call Waiting, set($Cwa,1)
Code16	Default = Disable Call Waiting Star Code	*57, Disable Call Waiting, set($Cwa,0)
Code17	Default = Do Not Disturb Star Code	*78, Do Not Disturb, set($Dnd,1)
Code18	Default = Disable Do Not Disturb Star Code	*79, Disable DND, set($Dnd,0)
Code19	Default = Repeat Dial Star Code	*66, Repeat Dial, rpdi($Ldn)
Code20	Default = Disable Repeat Dial Star Code	*86, Disable Repeat Dial, rpdi
Code21	Default = Set Speed Dial Star Code	*74(x|xx), Set Speed Dial, coll($Spd[$Code])
Code22	Default = Check Speed Dial Star Code	*75(x|xx), Check Speed Dial, say($Spd[$Code])
Code23	Default = Loopback Media Star Code	*03, Loopback Media, set($Lbm1,1)
Code24	Default = Loopback RTP Star Code	*04, Loopback RTP Packet, set($Lbp1,1)
Code25	Default = Force G711u Codec Star Code	*4711, Use G711 Only, set($Cdm1,3)
Code26	Default = Force G729 Codec Star Code	*4729, Use G729 Only, set($Cdm1,4)
Code27	Default = Clear Speed Dial Star Code	*76([1-9]|[1-9]x), Clear Speed Dial, set($Spd[$Code],)
Code28	Default = Easy WiFi Setup Mode	*27, Easy WiFi Setup, set($wifi,1)
Code29	Default = Barge In Star Code	*96, Barge In, set($Bar1,1)
Code30	Default = Make OBiBluetooth discoverable (for 2 minutes)	*28, OBiBT Discoverable, btdscvr(0)

User Settings

User Preferences Web Page

Parameter	Description	Default Setting
User Preferences		
Language	Set the language for the on-screen messages	English-US
TimeFormat	Choose the time display format on the the title bar of the phone screen. Valid choices are: • default • 1:35 pm • 13:35 • 1:35	default
DateFromat	Choose the date display format on the title bar of the phone screen. Valid choices are: • default • 1/31/2014 • 1/31/14 • 1/31 • 31/1/2014 • 31/1/14 • 31/1 • Jan 31, 2014	default
AutoHomePage	Choose the delay to automatically return the screen to the HOME (idle) screen.	1 minute

	Valid values are: • disabled • 30 seconds • 1 minute • 10 minutes • 1 hour • 4 hours	
Skin	Choose the skin (or style and color scheme) of the user interface. Valid choices are: • Tomáš • Colorful • Ulrik	Tomáš
BackgroundPicture	A URL (HTTP or HTTPS) to a picture file or the path name of an interally stored picture.	Default
DefaultRingtone	A URL (HTTP or HTTPS) to a ringtone file (.wav or MP3 format) or the path name of an internally stored ringtone file	/data/ringtones/Office A.wav
CallParkRingtone	A URL (HTTP or HTTPS) to a ringtone file (.wav or MP3 format) or the name of an internally short ring: `ding` or `blop`. It may be followed by two optional comma-separated parameter={value} pairs: - duration={*value*} . {*value*} is total duration to play in seconds; must be > 0; 0 means play forever. Default is 60 - interval={*value*}. {*value*} is interval between ring bursts in seconds; must be > 0. Default is 10	ding,interval=10,duration=60
DefaultFont		ptsans
HomeAppColumns	Number of columns to arrange the items in the Main Menu on the Home Screen. Valid values are 1, 2 or 3	3
PopUpMoreSoftKeyOptions	If enabled, show more softkey options in a popup window when user presses the “More” softkey whern there are more than 4 options. Otherwise, show softkey options in a circular list when user presses the “More” softkey	true
PackCallsOnDisplay	This option is applicable when one ofr more Call Keys (feature key that is assigned the Call Appearance function) have the MaxCalls parameter set to greater than 1. If the option is disabled, calls controlled by different Call Keys are shown on separate screens (as the user scrolls through the calls). Otherwise, calls controlled by different Call Keys may be	true

	shown on the same screen (packed together)	
LineKeyTabs	This option controls whether to show multiple tabs of (virtual) Line Keys on the screen	true
LineKeyMaxTabs	Must be either `2` or `4`. Appliable only if LineKeyTabs = `true`	4
LeftLineKeyTabs	This option controls whether to show multiple tabs of (virtual) Left Line Keys on the screen. Available on OBi2000 series ony.	true
LeftLineKeyMaxTabs	Must be either `2` or `4`. Appliable only if Left LineKeyTabs = `true` Available on OBi2000 series ony.	4
DimScreenDelay	If greater than 0, this is the delay in seconds before dimming the screen when there are no activities. Dimming is done by setting the LCD brightness to the minimum value of 0. Any key press or new call will restore normal LCD brightness. This operation is independent of Screen Saver settings	0
ToggleCallsApp	Show the Calls App when pressing the HOME key on the HOME screen where there are active calls. Press the HOME key again to re-hide the Calls App.	false
ScreenSaver	Tune on/off the screen saver features	false
ScreenSaverDelay	Number of seconds of inactivity before starting screen saver	10
ScreenSaverType	Choose the type of screen saver to use: • Slide Show • Slide Show (User Pictures Only) • Slide Show (Specified URL Only) • Turn Off Display	Slide Show
ScreenSaverPictureURL	When **SceenSaverType** is `Slide Show (Specified URL Only)`, this parameter specifies the URL to get the pictures for the slide show. It can be a http:// or https:// url to get the screen save picture(s), or the absolute path of an internal folder to get the screen saver pictures. If http url is used, the server may indicate 'no-cache' in the Cache-Control header of the http response to the phone to force the phone to download a fresh picture from the server on every screen picture refresh. You can use $count macro in the http/https URL which is expanded into the number of refreshes when the URL is executed. It starts with 1, 2, 3, …, max-	

	count and the recycles. The max-count value can be specified as a comma-separated attribute in this URL also; the default is 0x7fffffff. For example: http://xyz.com/pic-$count.png,max-count=4 http://xyz.com/pic-${count}.png,max-count=15	
SlideShowInterval	The interval to refresh each picture in a slide show when screen saver is running	5
ScreenSaverLock	Control whether dismissing the screen saver requires a passcode	false
UnlockPasscode	Passcode to unlock the screen saver	
ScreenSaverShowCustomContents	Show the custom contents defined in the parameter 'ScreenSaverCusomContents' (under the Screen Item Customization group), super-imposed on the current screen saver picture	true
Brigntness		6
PreferredAudioDevice		Speaker
PreferredHeadsetDevice		RJ9 Headset
RingerVolume		5
SpeakerVolume		10
HandsetVolume		5
HeadsetVolume		5
Headset35mmVolume		5
HeadsetBTVolume		5
MicGain		3
HandsetGain		3
HeadsetGain		3
Headset35mmGain		3
HeadsetBTGain		3
EqEnable		true
AecEnable		Non-linear
IncomingPageAlertTone	Choose whether to play a beep when the phone answers an incoming Intercom/Page call. Choices are: • none • beep	beep
UseEnhanceDialer	Enable enhanced dialer where a list of recently dialed number that partially match the currently entered digits is displayed below the dialer input box; user can highlight an entry in the list to make the call to	false
DefaultLineKeyTab	Valid Range: 1 – 4 (for OBi1032/1062 and OBi2000 Series) and 1 – 2 (for OBi1022)	1
DefaultLeftLineKeyTab	Valid Range: 1 – 4 (Available in OBi2000 Series Only)	1
HideMainMenu	Enable hiding of the main menu on screen by defaut; use press the HOME Key to	false

	toggle the Main Menu on/off	
HideLineKeys	Hide/Show Line Keys on screen by pressing the HOME key	false
ToggleMainMenuAndLineKeysTogether	Hide/Show Main Menu and Line Keys together by pressing the HOME key	false
ToggleCallsApp	Hide/show the Calls app when pressing the Home Key when there is one or more calls.	false
CustomMainMenu	Use customized Main Menu	false
AutoAnswerAutoMute	Enable auto-mute when the phone auto-answers an incoming Intercom/page call	false
AutoAnswerOffWhenBusy	Automatically turn off auto-answer intercom calls when the phone is busy (if the user is making a call or talking on a call).	false
PreferredBuddyListAction	The default action when user clicks an item on the Buddy List. Choose among: call, text (Available in OBi2000 Series Only)	call
Phone Book Fields		
Name		true
Number		true
Service		true
Ringtone		true
Picture		true
FirstName		false
LastName		false
Email		true
MobileNumber		true
OfficeNumber		false
HomeNumber		false
Address		false
Company		false
Group		false
Phone Book Settings		
Name	The label to display for this item on the phone's main menu screen	Contacts
ActionURL	The (HTTP or HTTPS) URL to execute when the user selects the Phone Book item from the phone's menu, if one is specified. Otherwise, the phone launches the built-in Phone Book application.	
Groups	A comma sepeated list of group names to group contacts within the built-in phone book	Co-Workers, Customers, Family, Friends

Speed Dials Web Page

Parameter	Description	Default Setting
N (Repeated for N = 1 – 99)	Speed Dial *N*	

User Defined Digit Maps Web Page

For an explanation of this feature, please refer to the section User Defined Digit Maps under Digit Map Configuration.

Parameter	Description	Default Setting
User Defined Digit Map 1		
Label	A 2-16 characters long label to reference this digit map in other digit maps and call routing rules. It must be alphanumeric, not containing any spaces, and different from other user-defined or built-in digit map labels	ipd
DigitMap	A valid digit map	(xx.<*:@>xx?x?<*:.>xx?x?<*:.>xx?x?<*:.>xx?x?|xx.<*:@>xx?x?<*:.>xx?x?<*:.>xx?x?<*:.>xx?x?<*::>xx?x?x?x?) Note: this default value supports IPv4 dialing
User Defined Digit Map *N* (*N* = 2 – 10)		
Label	A 2-16 characters long label to reference this digit map in other digit maps and call routing rules. It must be alphanumeric, not containing any spaces, and different from other user-defined or built-in digit map labels	
DigitMap	A valid digit map	