



## A Tutorial on Digit Maps and Call Routing for OBi Devices

**Version 1.1**

December 2014

## TABLE OF CONTENTS

INTRODUCTION .....	3
TERMINALS ON THE OBI .....	4
DIGIT MAP DESIGN .....	5
DIGIT MAP FORMAT .....	6
DIGIT MAP ELEMENTS.....	6
DIGIT MAP RULE EXAMPLES .....	11
COMPLETE DIGIT MAP EXAMPLES .....	12
NORTH AMERICA (DEFAULT) .....	12
NEW ZEALAND.....	12
AUSTRALIA.....	12
GREAT BRITAIN AND NORTHERN IRELAND.....	12
REPUBLIC OF IRELAND .....	13
FRANCE .....	13
SECONDARY DIAL TONES .....	13
USER DEFINED DIGIT MAPS .....	14
OUTBOUND CALL ROUTES.....	15
OUTBOUND CALL ROUTE EXAMPLES .....	16
INBOUND CALL ROUTES .....	17
INBOUND CALL ROUTE EXAMPLES.....	18
ROUTING CALLS BETWEEN DEVICES.....	19

## **Introduction**

This tutorial is designed to give the reader a clear understanding of how to create and apply digit maps and call routing rules on OBi devices. This guide applies to all Obihai devices.

The examples contained within this document use number schemes from different countries to help give international readers better context when designing their own digit maps and call routes.

The changes to settings explained within this document can either be made via the device's local UI or via OBiExpert on the OBiTALK portal.

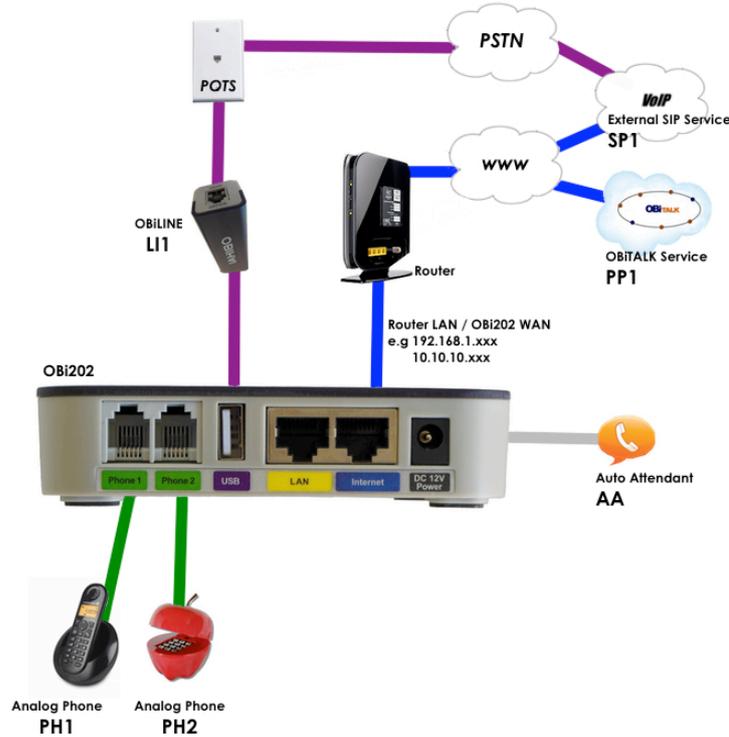
Visit <http://www.obitalk.com/forum/index.php?topic=61> to find out more about the two management methods.

Upon reading this document the reader should be able to successfully create digit maps for their required deployment and route calls between the voice terminals on the OBi as well as between multiple OBi devices.

*Obihai Technology, Inc.  
December 2014*

## Terminals on the OBi

Each physical connection and virtual service on the OBi device is addressed as a logical terminal by the software. The diagram below shows how the OBi addresses each of these terminals:



OBi devices support the following terminals:

Interface/Service:	Terminal Label(s):
Phone (FXS) Port(s) - ATA Handset - IP Phone	PH1, PH2 ... PHn
Line (FXO) Port(s)	LI1, LI2 ... LIn
IP Voice Service(s)	SP1, SP2 ... SPn
Auto Attendant(s)	AA1, AA2 ... AAn
Bluetooth Device(s)	BT1, BT2
Voice Gateway(s)	VG1, VG2 ... VGn
Trunk Group(s)	TG1, TG2 ... TGn

Terminal names when referenced on their own are not case sensitive. You will notice variations of case throughout this document to highlight this behaviour. Terminals interact in a matrix, with call flow between terminals defined by the inbound and outbound call route for each terminal that participates in the call flow:

	SP1 Service	SP2 Service	SPn Service	Line Service	OBiTALK Service	Bluetooth Service
SP1 Service	yes	yes	yes	yes	yes	yes
SP2 Service	yes	yes	yes	yes	yes	yes
SPn Service	yes	yes	yes	yes	yes	yes
Line Service	yes	yes	yes	yes	yes	yes
OBiTALK Svc	yes	yes	yes	yes	yes	yes
Bluetooth Svc	yes	yes	yes	yes	yes	no

## Digit Map Design

The easiest way to get started is by listing all the number types you wish to cater for in your digit map. List these in a table or spread sheet for easy reference when creating your custom digit map.

The table below shows an example for a few Australian number types:

Description	Number Dialed
<b>Local Numbers in Sydney, NSW:</b>	
8 digits long, starting with any digit between 3 and 9	3xxx xxxx 4xxx xxxx 5xxx xxxx 6xxx xxxx 7xxx xxxx 8xxx xxxx 9xxx xxxx
<b>National Numbers:</b>	
01 xxxx xxxx Australian satellite-based phone numbers	01 xxxx xxxx
02 xxxx xxxx NSW, ACT	02 xxxx xxxx
03 xxxx xxxx VIC, TAS	03 xxxx xxxx
04xx xxx xxx Mobiles	04xx xxx xxx
05 xxxx xxxx Location Independent	05 xxxx xxxx
07 xxxx xxxx QLD	07 xxxx xxxx
08 xxxx xxxx SA, NT, WA	08 xxxx xxxx
<b>International Numbers:</b>	
0011 numbers of variable length for International access	0011xxx.....

Now you have a list of the number forms you wish to be able to quickly dial, we can move on to creating the rules needed to match and route each type of number.

## Digit Map Format

A digit map can be used to perform the following tasks:

- Ensure a complete number is dialed
- Transform dialed digits
- 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)**

Note that these two rules are the same; the difference is simply the spaces that have been added for clarity. 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:

<p><i>Any combination of:</i> 0-9 * # + - A-Z a-z <i>Excluding:</i> m M s S x X</p>	<p><b>Literals</b></p> <p>Use literals to explicitly match a string. Literals are used to exactly match a number or parts of a number. They can also match alphanumeric characters, with the following exclusions: m, M, s, S, x, X as these characters have special meaning within the digit map.</p>
<p><i>Example:</i> 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: <b>12125557722</b> or <b>1 212 555 7722</b> as spaces may be added for clarity</p>	

' '	<p><b>'Quoted Literals'</b></p> <p>Everything inside a pair of single quotes is treated as a literal except for the single quote ' character itself.</p> <p><i>Example:</i></p> <p>To explicitly match the SIP URI matt@sip-service.com we need to include '' as the address includes reserved characters. We would write our matching rule as:</p> <p><b>'matt@sip-service.com'</b></p>
-----	--

x	<p><b>x Wildcard</b></p> <p>The "lowercase x" - <b>x</b> - wildcard matches any digit from 0-9. It is important to note that <b>x</b> is CASE SENSITIVE.</p> <p><i>Example:</i></p> <p>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:</p> <p><b>1 706 xxx xxxx</b></p>
---	---

.	<p><b>. Matching Function</b></p> <p>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.</p> <p><i>Example:</i></p> <p>The expression <b>xx.</b> 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:</p> <p><b>011 64 xx.</b>      <i>Using US international dialing notation</i></p> <p><b>0011 64 xx.</b>     <i>Using Australian international dialing notation</i></p> <p><b>00 64 xx.</b>        <i>Using Standardized international dialing notation</i></p>
---	--

[ ]	<p><b>Set</b></p> <p>"Square brackets" - [ ] - enclose a set of numerals and/or characters that are used to match a single digit or character. 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]</p> <p><i>Example:</i></p> <p>Let's look at specifically matching the numbers 1,2,5,6,7 or 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:</p> <p><b>020 [378]xxx xxxx</b></p> <p>This rule says, match 020 then match either 3 or 7 or 8 followed by 7 digits</p>
-----	---

[ ^ ]	<p><b>Exclusion Set</b></p> <p>An exclusion set includes a leading caret, or "hat", within a set of "square brackets" - [ ^ ]</p> <p>An exclusion set matches any single alphanumeric character that is <u>not</u> within the set.</p> <p><i>Example:</i></p> <p>To match any arbitrarily long sequence of digits that does not start with * we would write our matching rule as follows:</p> <p><b>[^*]xx.</b></p> <p>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 <b>[3-9]xxx xxxx</b> or using an exclusion set we could write:</p> <p><b>[^0-2]xxx xxxx</b></p>
-------	--

!	<p><b>! Call Bar</b></p> <p>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 <i>barring rule</i>.</p> <p><i>Example:</i></p> <p>If we wish to bar all calls to 1900 numbers (regardless of length) we could use the following rule:</p> <p><b>!1900xx.</b></p>
---	--

< elements : literals >	<p><b>Element to Literal Transformation</b></p> <p>An element-to-literal transformation allows us to substitute the digit sequence matching <i>elements</i> with the given <i>literals</i>. The expression is contained within a set of "pointy brackets" - &lt; &gt; - and elements are separated from literals using a colon :</p> <p><i>Usage Notes:</i></p> <ul style="list-style-type: none"> <li>• Single quote ' ' syntax is NOT needed or allowed for the <i>literals</i> in this context; special characters may be used here as they do not apply in this context either.</li> <li>• 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.</li> <li>• The literals part can be empty also but the colon - : - MUST NOT be omitted. This case is useful for removing part of dialed digits. <i>Elements</i> and <i>literals</i> MUST NOT both be empty.</li> </ul> <p><i>Example:</i></p> <p>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:</p> <p><b>&lt;112:000&gt;</b> take 112 and replace it with 000</p> <p>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:</p> <p><b>&lt;02:&gt;xxxx xxxx</b> take 02, replace it with nothing then match the next 8 digits</p> <p>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:</p> <p><b>&lt;02&gt;xxxx xxxx</b> or <b>&lt;:02&gt;xxxx xxxx</b></p>
-------------------------	---

<b>S, S0, S1 ... S9</b>	<p><b>Digit Timer</b></p> <p>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 - <b>S</b> - is CASE SENSITIVE.</p> <p><i>Example:</i></p> <p>The notation <b>S0, S1, S2, S9</b> gives digit timer values of 0, 1, 2 and 9 seconds respectively; <b>S</b> is equivalent to <b>S1</b>; <b>S0</b> is the same as "blank". You can concatenate multiple <b>S</b> elements together if you need more than 9s timeout, such as <b>S9S5</b> for a 14s timeout. To create a hotline to the number 1-408-890-6000 we would write:</p> <p style="text-align: center;"><b>&lt;S0:14088906000&gt;</b></p>
-------------------------	--

The next two elements, *(map)* and *(Mlabel)*, imply that the OBi digit maps are *recursive*. 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.*

<b>(map)</b>	<p><b>Embedded Digit Map</b></p> <p>An embedded digit map contains elements within a pair of brackets - <b>( )</b> - that are used to match subsequent digits.</p> <p><i>Example:</i></p> <p>To match any number that starts with *74, followed by 1 or 2 digits we would write our matching rule as follows:</p> <p style="text-align: center;"><b>*74(x xx)</b></p>
--------------	---

<b>(Mlabel)</b>	<p><b>Named Embedded Digit Map</b></p> <p>A named embedded digit map is used for matching subsequent digits, where <i>label</i> refers to one of abbreviated terminal names or a user defined digit map label. As an example, <b>(Msp1)</b> refers directly to the Digit Map contained within <b>Voice Services → SP1 Service</b> of the device admin webpage.</p> <p><i>Example:</i></p> <p>Let's consider one of the OBi's default digit map rules that directs a call preceded 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:</p> <p style="text-align: center;"><b>**2(Msp2)</b></p>
-----------------	--

The following advanced elements allow you to further compact your rules for more elegant digit map syntax:

<b>X</b>	<b>X Wildcard</b> The "uppercase x" - <b>X</b> - 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 <b>X</b> is CASE SENSITIVE. <i>Usage Notes:</i> <ul style="list-style-type: none"><li>• The element <b>X</b> is equivalent to [x*] or [0-9*x]</li></ul> <i>Example:</i> If we wish to catch all three digit numbers including star codes, we could write our matching rule as follows: <b>XXX</b>
----------	---

<b>@</b>	<b>@ Wildcard</b> The "at symbol" - <b>@</b> - wildcard matches <u>any</u> alphanumeric character except # <i>Example:</i> To match any arbitrarily long alphanumeric sequence (except #) that does not start with * we would write our matching rule as follows: <b>[^*]@.</b>
----------	---

<b>?</b>	<b>? Matching Function</b> The "question mark" - <b>?</b> - 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 <i>Example:</i> The expression <b>xxxx?</b> 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: <b>xxxxx?</b>
----------	--

## 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	<b>1 408 xxx xxxx</b>
Match any 7-digit number and prepend 1408 to the number when making the call	<b>&lt;1408&gt; xxx xxxx or &lt;:1408&gt; xxx xxxx</b>
Match any number that starts with 011 followed by one or more digits	<b>011xx.</b>
Add '+' to any 11-digit number that starts with 1	<b>&lt;+&gt;1xxxxxxxxxx</b>
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	<b>&lt;**1:&gt;020 [378]xxx xxxx</b>
Create a hotline to 1234	<b>&lt;:1234&gt; or &lt;S0:1234&gt;</b>
Create a warm line to 1234 that will be called if the user doesn't enter any digits within 4 seconds	<b>&lt;S4:1234&gt;</b>
Match any number with at least 8 digits that ends with 8537683, such as 1 510 853 7683, 9 853 7683	<b>xx.853 7683</b>
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	<b>@. 408 890 6000</b>
Add a # to the end of any number with 1 or more digits	<b>xx.&lt;#&gt;</b>
Bar calls to premium rate numbers beginning with 084, 087 and 09	<b>!08[47]x. and !09x.</b>

Now we will create an example 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 I have used spaces in the digit map. It is fine to include spaces to help make your digit map more readable.

## Complete Digit Map Examples

The following examples are complete digit maps for each of the countries listed. You can use them as a starting point to create your own digit map.

It's worth noting the effect that legacy numbering platforms can have on the digit map that is required to successfully match all common numbers dialed within a country. The North American Numbering Plan (NANP) covers a total of 25 countries and proves to yield the most efficient digit maps for use by the OBi, whereas New Zealand's digit map is quite large for a country of small population.

In the case of NZ, there are a large number of mobile number length variations (numbers starting with 02) that include network-operator-specific digits. These digit combinations used in this approach make it very easy for the user to identify another party's mobile phone carrier and in some cases even identify the phone's contract type via the number.

As an example, take a mobile phone in NZ starting with 021-0. This number was originally assigned by Vodafone (or BellSouth up to 1998), this is indicated by the prefix 021. The 0 identifies that this number was originally assigned to a device on a pre-pay (or pay-as-you-go) package. Although this numbering system affords a more intuitive experience for the end user, it requires a more complex digit map to be implemented on the OBi.

*Note: If you wish to copy and paste these digit maps, don't paste them directly to your device. Paste the digit map into a text editor first (Notepad, Text Wrangler etc) and remove the line breaks so the text is in a single row. You can now copy this single line of text and paste it into the OBi's digit map field*

### North America (Default)

North American Numbering Plan (NANP) for USA, Canada and 23 other countries

```
(1xxxxxxxx|<1>[2-9]xxxxxxxx|011xx.|xx.|(Mipd)|[ ^*#]@@.)
```

### New Zealand

```
(111|[2-9]xx xxxx|0[34679]xxx xxxx|02[0279]xxx xxxx|021 0xxx xxxx|  
021 2xx xxxx |021[3-9]xx xxx|024 0x.|024[1-9]xxx xxxx|028[0134567]xx xxxx|  
028[289]xxx xxxx|026[1-3]xx xxx|0264 xxx xxx|0508 xxx xxx|080[0-8]xx xxxx  
|08[2-3]xxx|01681x.|08322|014xx|015xxx|017[02]|0900 xx xxx|00xxxx.|010|018|  
137|195[67]|511|08320123[124] |1xxx|0197xx.|00xx.|xx.|+xx.|(Mipd)|[ ^*#]@@.)
```

### Australia

```
(000|xxxxxxxx| 0[1234578]xxxxxxxx|1800 xxx xxx|180 xxx|1100|114xxxx|  
119x|12[23]x|124xx|125xxx|1268x|1268xxx|1268xxxx|127x.|1282|128xxx|  
15x.|190xxxx|1300xxxx|13xxx|0011xx.|xx.|+xx.|(Mipd)|[ ^*#]@@.)
```

### Great Britain and Northern Ireland

```
(999|11[12]|101|116000|116111|116123|1471|1571|0[15]xxxxxxxx|?  
0[27]x xxxx xxxx|0800xxx xxxx|0808xxx xxxx|08001111|08[47]x xxx xxxx|  
0845464x|03xx xxx xxxx|118xxx|100|155|195|09xx xxx xxxx|00xx.|xx.|+xx.|(Mipd)|[ ^*#]@@.)
```

## Republic of Ireland

```
(112|999|[2-9]xxxxxx|023[2-9]xxxxxx|02[24-9][2-9]xxxx|01[2-9]xxxxxx|
118xx|04[0-79]xxxxxxx|0[5679]xxxxxxx|0818xxxxxx|08[2-9][0-46-9]xxxxxx|
08[2-9]5xxxxxxx|18xxxxxxx|15xxxxxxx|048xxxxxxx|19xx|171|14x.|
00x.|xx.|+xx.|(Mipd)|[^*#]@@.)
```

## France

```
(11[245]|1[578]|119|19[16]|11[68]xxx|10xx|16xx|
0[1-9]xx xx xx xx|00xx.|xx.|+xx.|(Mipd)|[^*#]@@.)
```

## Secondary Dial Tones

You can tell the OBi to play a different 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))
```

This tells the device to play the "Second Dial Tone" **di2** when **\*\*1** is dialed, or play the "Outside Dial Tone" **od** when **\*\*8** is dialed.

Here is a full list of acceptable (case insensitive) values of <tone> :

Value	Played 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

## 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. The first is occupied by **ipd** (referenced as **(Mipd)**) that deals with IP dialing (see admin guide for detail). You can choose whether to leave this in-situ or replace it. If you replace it, then make sure you delete reference to it in any other active digit maps on the device.

Each user defined digit map is specified with two parameters:

**Label:** This is the name you refer to the digit map with. Let's say you want to create a digit map to identify numbers made to London, as dialed within the UK. You could make the digit map label **london** and would then refer this digit map using **(Mlondon)** from any digit map on the OBi.

**Digit Map:** This is where you enter your digit map. In the case of London, this digit map would be **(020 [378]xxx xxxx)**

As an example, the numbering plans of many countries can be grouped into the following categories:

Call Type	Digit Map Label	Digit Map Reference
Calls within our area code	local	(Mlocal)
National Calls	nat	(Mnat)
Mobile Calls	mobile	(Mmobile)
International Call	int	(Mint)
Emergency Numbers	sos	(Msos)
Free-call Numbers	free	(Mfree)
Local-Rate Special Numbers	lrsn	(Mlrsn)
Special Numbers	special	(Mspecial)
Premium-Rate Numbers	prem	(Mprem)

User defined digit maps allow you to craft recursive rules that can be re-used within the digit maps of terminals on the device. In the case of Australia, we have documented how to place the entire numbering plan within user defined digit maps to enable a granular and efficient approach to single stage dialing. We can apply the information in this tutorial to your own digit map, regardless of your country. If you wish to explore this further you can find the *OBi VoIP Device Australian Localisation Workbook* at [www.obihai.com/docs-downloads](http://www.obihai.com/docs-downloads)

## Outbound Call Routes

We use a digit map to match or transform a number or URI that's been entered. Once the digit map has matched the entered sequence, we can then use the outbound call route to direct the call to the correct terminal on the OBi. The outbound call route directs the call from the perspective of the user placing the call from the OBi device.

We can consider the form of our rules to follow the following structure:

Outbound Call Route Structure		
{rule},{rule},{rule},{rule}		
rule =		
{callee-list	:	terminal}
callee-list =		terminal =
{callee callee	:	PHx, AAx, SPx, Llx, or PPx}
callee =		
number OR (digit-map) OR @		(@= <i>any number</i> )

There are a few things we need to keep in mind when designing call routes:

- Within the **OutboundCallRoute** parameter, every rule must be placed inside a pair of { } "curly brackets"
- Rules are tested from left to right
- The first matched rule will be selected
- Only one terminal is supported per rule
- If there is only a single rule, then { } is optional
- **{0255556666:SP2}** is the same as **{{0255556666}:SP2}** - as 02 5555 6666 consists only of literal elements, the parentheses are optional
- If the rule includes things like xxxx and so forth, then round brackets ( ) are also required, like this: **{{(025xx.):SP2}**

## Outbound Call Route Examples

Function	Outbound Call Route
Send 999 emergency calls via the PSTN using the OBi's Line <b>li</b> terminal	<b>{999:LI}</b>
Send all calls to Japan via the <b>SP3</b> terminal. We will use the US international dial code of 011 and Japan country code 81	<b>{{(01181xx.):SP3}</b>
Send mobile phone numbers within the user defined digit map ( <b>Mmobile</b> ) via the Bluetooth <b>BT</b> terminal	<b>{{(Mmobile):BT}</b>
Send local numbers defined in the user defined digit map ( <b>Mlocal</b> ) and numbers beginning with 020 via the <b>SP1</b> terminal	<b>{{((Mlocal)020xx.):SP1}</b>

When combined with user defined digit maps, the outbound call route provides a powerful way to direct outbound calls while maintaining a very simple mode of user operation.

When testing your outbound call route, rather than delete the existing call route in the OBi, add your rule to the start of what already exists. As an example, let's add the call route example for mobile calls from the table above to the default outbound call route for the OBi IP Phone **PH** terminal:

```
{{(Mmobile):BT},{{(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}
```

It's important to remember, rules are tested left to right, and you need to ensure you rule does not conflict with an existing call route rule.

Once you are confident with creating call routes, you can craft a fully customized set of outbound call routes to replace the factory default.

## Inbound Call Routes

Inbound call routes operate in the opposite direction to outbound call routes. The inbound call route directs inbound calls from the perspective of the call arriving at a terminal on the OBi device. By default, all terminals have their inbound call route set as **ph** which will send the inbound call to the phone terminal. However we may want to implement more advanced behavior to direct different types of inbound calls and to block others.

We can consider the form of our rules to follow the following structure:

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 Llx(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 > digit-map or number-to-call (the outbound CallerID)

As with Outbound Call Routes, the same usage criteria apply for Inbound Call Routes:

- Within the **InboundCallRoute** parameter, every rule must be placed inside a pair of { } "curly brackets" unless there is a single rule (i.e **ph**)
- Rules are tested from left to right
- The first matched rule will be selected
- Only one terminal is supported per rule
- If there is only a single rule, then { } is optional
- **{0255556666:SP2}** is the same as **{{0255556666}:SP2}** - as 02 5555 6666 consists only of literal elements, the parentheses are optional
- If the rule includes things like xxxx and so forth, then round brackets ( ) are also required, like this: **{{025xx.}:SP2}**

## Inbound Call Route Examples

In each of the examples below, you would change the call route for the inbound voice service that you wish to route calls for.

Function	Inbound Call Route
Direct inbound calls from anonymous callers to the Auto Attendant <b>AA</b> terminal	<b>{{(?):AA}</b>
Transform inbound international numbers with the prefix 00 (European number style) to remove the 00 and replace it with a +	<b>{{(&lt;00:+&gt;xx.):ph}</b>
Direct all inbound OBiTALK calls to the phone number 1-602-555-9887 via terminal SP1	<b>{@:SP1(16025559887)}</b>
Forward incoming local numbers contained in the user defined digit map ( <b>Mlocal</b> ) to the OBiTALK number 600555600 on the <b>PP</b> terminal	<b>{{(Mlocal):PP(600555600)}</b>

When testing your inbound call route, you can leave the **ph** terminal within the existing inbound call route (if you wish the phone to still ring), just surround it with a set of "curly brackets" **{ }** like this **{ph}**

Using the above example of directing inbound OBiTALK calls, under the InboundCallRoute for the OBiTALK voice service we would write:

**{@:SP1(16025559887)}**

Note, in this example, we have left out the ph terminal. If we were to leave ph in the route like this: **{@:SP1(16025559887)},{ph}** the phone wouldn't ring as the rule has already matched the inbound call. If we want to fork the call so the inbound call makes the OBi ring as well as the remote party, then we can edit our rule as follows:

**{@:SP1(16025559887),ph}**

Once you are confident with creating call routes, you can craft a fully customized set of outbound call routes to replace the factory default.

## Routing Calls Between Devices

Let's say you have multiple OBi devices on your network and want to share Terminals between devices. On the OBi we can use inbound and outbound call routes to direct calls out via a terminal on another OBi. This is called single stage dialing.

For this section we will explain how to do this with the OBiTALK protocol. However you can also achieve this using SIP if you wish to include 3<sup>rd</sup> party devices. We won't cover SIP routing in this section, however it is covered in the document *OBi Legacy IP Phone Workbook* at [www.obihai.com/docs-downloads](http://www.obihai.com/docs-downloads)

Take a sample network where we have an OBi1032 IP Phone and an OBi202 ATA with an OBiLINE connected to it to allow access to the PSTN. In this case, we are going to direct local calls made on the OBi1032 IP Phone out via the OBi202's **LI** terminal. On the OBi1032 we are going to setup a **Voice Gateway** that points to the OBi202's OBiTALK **PP** terminal. On the OBi202 we are going to direct the inbound call received on it's **PP** terminal to it's **LI** terminal.

On the OBi1032 visit **Voice Services → Gateways and Trunk Groups** and define a Voice Gateway to point to the OBi202. Our OBi202 has the (fictional) OBi Number 202555202, so we set the **AccessNumber** field as:

**PP(ob202555202)**

Next on the OBi1032 we need to define what constitutes a local call and how to deal with it after it has been entered. In this example we are going to use the example of local numbers dialled within London. These numbers are 8 digits long and can start with 3, 7 or 8. The digit map rule to match these numbers will be:

**[378]xxx xxxx**

Enter this rule at the start of your digit map for the **PH** terminal. Make sure you keep it within the correct digit map structure - for example ([378]xxx xxxx| ... | ... )

Now, we need to direct how to send these calls from the **PH** terminal using its outbound call route. We need to define the numbers to capture and where to send them. In this case we want to send it to the OBi202 with the (fictional) OBi Number 202555202. As such, our outbound call route for the OBi1032's **PH** terminal will read:

**{{[378]xxx xxxx}:vg1}**

Add this value to the start of the Outbound Call Route for the **PH** terminal.

Next on the OBi202 we need to define how to deal with the inbound call. In this case our OBi1032 has the fictional OBi number 600555600.

**{600555600>(Mli):li}**

Now when the OBi202 receives an inbound call from the OBi1032 it will take the dialed digits, match them with the digit map for the **LI** terminal and then send the call out via the **LI** terminal.