



**OpenVox Communication Co.Ltd**

***OpenVox-Best Cost Effective Asterisk Cards***

Address: F/3, Block No.127, Jindi Industrial Zone,  
Shazui Road, Futian District, Shenzhen, Guangdong 518048, China  
Tel:+86-755-82535461, 82535095, 82535362, Fax:+86-755-83823074

Business Contact: [sales@openvox.com.cn](mailto:sales@openvox.com.cn)  
Technical Support: [support@openvox.com.cn](mailto:support@openvox.com.cn)

Business Hours: 09:00-18:00(GMT+8) from Monday to Friday  
URL: [www.openvox.cn](http://www.openvox.cn)

***Thank You for Choosing OpenVox Products!***

## **Copyright**

Copyright© 2011 OpenVox Inc. All rights reserved. No part of this document may be reproduced without prior written permission.

## **Confidentiality**

Information contained herein is of a highly sensitive nature and is confidential and proprietary to OpenVox Inc. No part may be distributed, reproduced or disclosed orally or in written form to any party other than the direct recipients without the express written consent of OpenVox Inc.

## **Disclaimer**

OpenVox Inc. reserves the right to modify the design, characteristics, and products at any time without notification or obligation and shall not be held liable for any error or damage of any kind resulting from the use of this document.

OpenVox has made every effort to ensure that the information contained in this document is accurate and complete; however, the contents of this document are subject to revision without notice. Contact OpenVox to ensure you have the latest revision of this document.

## **Trademarks**

All other trademarks mentioned in this document are the property of their respective owners.



# Contents

<b>Chapter 1 Overview .....</b>	<b>4</b>
<b>Chapter 2 Software Installation .....</b>	<b>5</b>
<b>Chapter 3 Running chan_extra .....</b>	<b>10</b>
<b>Chapter 4 Configurations.....</b>	<b>11</b>
<b>Chapter 5 Functions.....</b>	<b>12</b>
<b>Chapter 6 Application Examples.....</b>	<b>14</b>
<b>Chapter 7 Virtual Serial .....</b>	<b>16</b>

# Chapter 1 Overview

Chan\_extra aims to both Asterisk and DAHDI application, it's running on Linux Operation System, including driver and application layers. It is using for driving OpenVox GSM products –G400P/G400E. Chan\_extra consists of three sessions: chan\_extra which is based on Asterisk, libgsmat shared library and opvvg4xx driver module. Here is a framework map on Asterisk, DAHDI and chan\_extra.

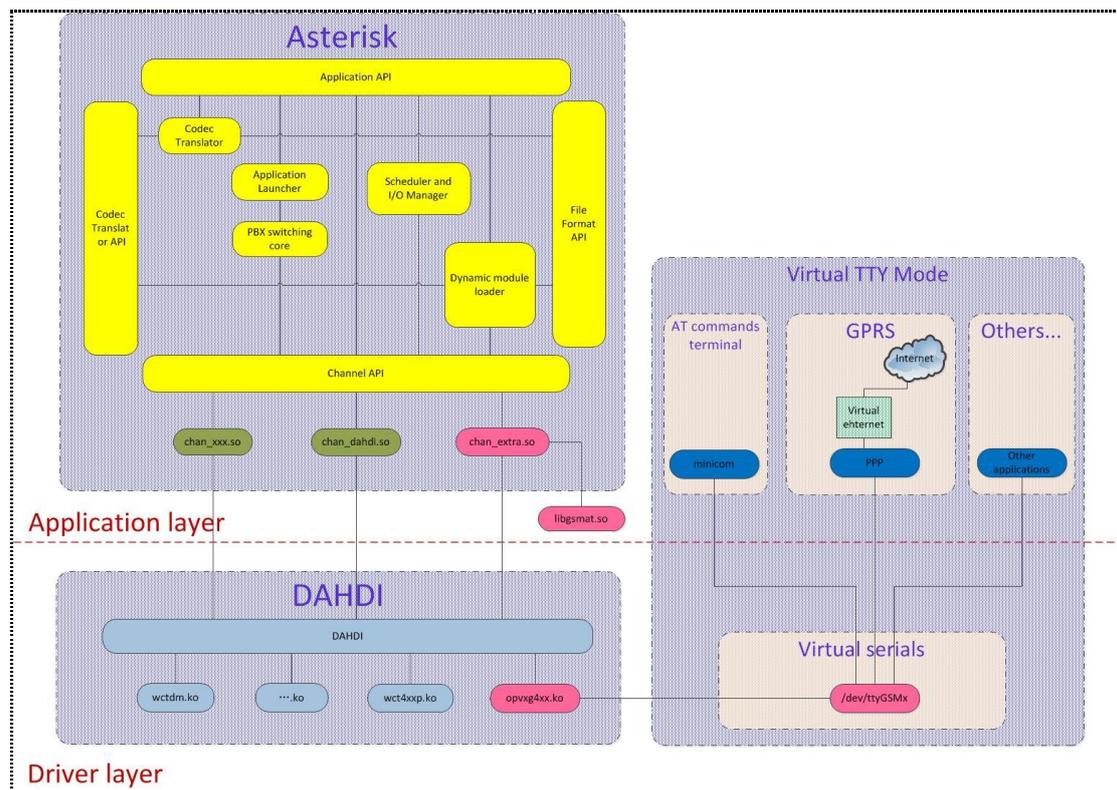


Figure 1 chan\_extra framework map



## Chapter 2 Software Installation

Installation conditions Chan\_extra is based on Linux Operation System. Currently it supports most of Linux distributions, including CentOS, RHEL, Debian, Ubuntu, and Fedora etc. Not only supports their 32-bit, but also 64-bit. Additionally, chan\_extra supports Elastix and Trixbox installation, users are able to manually select which one is suitable to their conditions.

### 1. Installation methods

There are three installation methods: source installation, Elastix installation and Trixbox installation. Source installation supports most of the Linux distributions. And Elastix installation supports versions of Elastix distributions. Meanwhile, Tixbox installation supports Trixbox distributions (only supports versions 2.8.0.3 and 2.8.0.4 so far).

### 2. Installation Steps

- a. Please go to website <http://downloads.asterisk.org/pub/telephony> and download Asterisk and dahdi-linux-complete packages, then decompress them to /usr/src. If you ignore this, the script installation script will help you later. But it's not as flexible as here.
- b. Please go to website <http://www.chan-extra.org/download/releases/> and download chan\_extra-2.0.5.tar.gz. Then decompress it to /usr/src/ by **`tar -xvzf chan_extra-2.0.5.tar.gz`** on the command line.
- c. Enter chan\_extra-2.0.5 and execute installation script install.sh. There are some illustrations to show you what the whole installation is.

The next illustration shows the beginning of the installation, the scripts will check if your current system has been installed some dependent packages that installation needed. And decide which version of packages will be installed.



```
#####
#                               OpenVox Extra Installation Script                               #
#                               v2.0.5                                                       #
#                               OpenVox Communication Co.,Ltd                               #
#                               Copyright (c) 2009-2011 OpenVox. All Rights Reserved.         #
#####

Checking for C development tools ... [ OK ]
Checking for C++ development tools ... [ OK ]
Checking for Make utility ... [ OK ]
Checking for ncurses library ... [ OK ]
Checking for ncurses-devel library ... [ OK ]
Checking for Perl development tools ... [ OK ]
Checking for Patch ... [ OK ]
Checking for bison... [ OK ]
Checking for bison-devel... [ OK ]
Checking for openssl... [ OK ]
Checking for openssl-devel... [ OK ]
Checking for gnutls-devel... [ OK ]
Checking for zlib... [ OK ]
Checking for zlib-devel... [ OK ]
Checking for kernel development packages... [ OK ]
Checking for libxml2-devel... [ OK ]

Press [Enter] to continue... █
```

**Figure 2 Checking dependent packages**

Press **[Enter]** will skip to the tty selection interface.

```
#####
#                               OpenVox Extra Installation Script                               #
#                               v2.0.5                                                       #
#                               OpenVox Communication Co.,Ltd                               #
#                               Copyright (c) 2009-2011 OpenVox. All Rights Reserved.         #
#####

Please Note:

  If you select 'y'(TTY mode), you will be able to directly operate GSM
  modem to use some features like GPRS dial-up access, virtualterminal
  over minicom, etc. By default, it doesn't recommend. It might cause
  some unexpected issues.

  If you select 'n', you will have a stable voice, SMS transmission system.

Do you need virtual TTY mode feature? (y/n) █
```

**Figure 3 TTY mode selections**

If you need virtual serial port features, please select **y**.

Once you select **y** or **n**, the system will select different compilation methods. But on the interface we are not aware. Press **y** or **n** you will enter installation interface; it will show you there are four options for you: Source installation, Trixbox installation, Elastix installation and Exit.



```
#####
#                               OpenVox Extra Installation Script                               #
#                               v2.0.5                                                       #
#                               OpenVox Communication Co.,Ltd                               #
#                               Copyright (c) 2009-2011 OpenVox. All Rights Reserved.         #
#####

1) Source code install
2) Trixbox-dahdi install
3) Elastix-dahdi install
q) quit

Please enter your selection (1..3 or q) ->|
```

**Figure 4 Selecting installation methods**

If you select [1], you will enter DAHDI installation interface, and the script will scan and check if there are DAHDI packages located there. If exist, they will be listed on the screen. If not, you will be able to download DAHDI from its official website. Also you can ignore this step, if you have installed DAHDI(including our opvngx4xx driver module of chan\_extra)in advance.

```
#####
#                               OpenVox Extra Installation Script                               #
#                               v2.0.5                                                       #
#                               OpenVox Communication Co.,Ltd                               #
#                               Copyright (c) 2009-2011 OpenVox. All Rights Reserved.         #
#####

Looking for dahdi-linux-complete directory in /usr/src ...

1 : /usr/src/dahdi-linux-complete-2.6.1+2.6.1
2 : /usr/src/dahdi-linux-complete-2.4.1.2+2.4.1
-----
n : Download and install dahdi-linux-complete-2.5.0+2.5.0.tar.gz [Default installation]
m : Enter dahdi-linux-complete dir path manually
d : Download the latest version of dahdi-linux-complete
q : Skip the step
(ctrl-c to Exit)
Please select working dahdi-linux-complete directory [1-2, n, m, d, q]:|
```

**Figure 5 DAHDI installation option**

Once you select your favorite option, DAHDI will be compiled and installed. If succeed, the script is going to enter Asterisk installation interface. Be the same with DAHDI, the script will scan and check if there are Asterisk packages in */usr/src/*. If yes, they will be list on this interface. Otherwise, you will be able to manually indicate where the Asterisk source code is. Of course, if you have done this, just skip it.

```
#####
#           OpenVox Extra Installation Script           #
#                   v2.0.5                             #
#           OpenVox Communication Co.,Ltd             #
#           Copyright (c) 2009-2011 OpenVox. All Rights Reserved. #
#####

Looking for Asterisk directory in /usr/src ...

1 : /usr/src/asterisk-1.8.13.0
2 : /usr/src/asterisk-1.8.11.0
-----
n   : Download and install asterisk-1.8-current.tar.gz [Default installation]
m   : Enter Asterisk dir path manually
q   : Skip the step
(ctl-c to Exit)
Please select working Asterisk directory [1-2, n, m, d, q]: █
```

**Figure 6 Asterisk installation option**

Since you select your Asterisk version yet, system will automatically compile and install it. And it will ask you to select installing default configuration files or not at the last step. If you are the first time to install it, select **y**. If not, select **n**.

```
+---- Asterisk Installation Complete -----+
+
+   YOU MUST READ THE SECURITY DOCUMENT   +
+
+ Asterisk has successfully been installed. +
+ If you would like to install the sample +
+ configuration files (overwriting any    +
+ existing config files), run:           +
+
+           make samples                   +
+
+----- or -----+
+
+ You can go ahead and install the asterisk +
+ program documentation now or later run:   +
+
+           make progdocs                   +
+
+ **Note** This requires that you have     +
+ doxygen installed on your local system   +
+-----+
WARNING WARNING WARNING

Your Asterisk modules directory, located at
/usr/lib/asterisk/modules
contains modules that were not installed by this
version of Asterisk. Please ensure that these
modules are compatible with this version before
attempting to run Asterisk.

    app_dahdiscan.so
    app_pickupchan.so
    res_indications.so

WARNING WARNING WARNING
Would you like to install the asterisk configuration files (y/n) █
```



### Figure 7 Configuration files selection

The whole installation process will be ended if you get the illustration below.

```
=====
Installing Asterisk ... [ OK ]
=====
```

Figure 8 Finishing installation process

- d. All the installations have been successfully finished if all the steps are correct above.
- e. Elastix and Tribox are most the same as source code installation, the main difference is last step, you have to note that it system will override the configuration files in /etc/asterisk, so if you select Elastix or Tribox installation, you must pay more attention on this step. We suggest you backing up default configuration files of Elastix and Tribox before trying this.



## Chapter 3 Running chan\_extra

As we know, chan\_extra is running on Asterisk and DAHDI environment, we have to load DAHDI driver and start Asterisk application to use chan\_extra.

Executing steps

- a. Execute **service dahdi start** or **/etc/init.d/dahdi/start** to start DAHDI driver.
- b. Execute **dahdi\_genconf** to generate file extra-channels.conf in **/etc/asterisk/**. This is a configuration file for G400P/E. Don't need to perform it anymore until you have new hardware upgrade.
- c. Execute **dahdi\_cfg -vvv**, the system will configure kernel in **/etc/dahdi/system.conf**. This step must be done after loading driver module.
- d. Execute **asterisk -vvvvvc** to start Asterisk application, you will see the interface like as below.(right here, there are two spans in a G400P)

```
Asterisk Ready.  
  == Parsing '/etc/asterisk/cli.conf':    == Found  
*CLI>    == D-Channel on span 1 up  
  == D-Channel on span 2 up
```

Figure 9 Asterisk starting interface

- e. You will be able to perform **gsm xxx** commands on the CLI once you start Asterisk successfully.

## Chapter 4 Configurations

In order to use `chan_extra`, we still have some steps to configure.

### 1. Configuring G400P/E.

The configuration file of G400P/E is `extra-channels.conf` in `/etc/asterisk`. It's automatically generated by `dahdi_genconf`. Here is a sample definition of one span in `extra-channels.conf`.

```
; Span 1: opvsg4xx/0/1 "Open Vox G400P GSM/CDMA PCI Card 0" (MASTER)
group=13 ; Group settings, you can assemble different spans to a group.
context=from-gsm-100 ; Context name, dialplan will search it when incoming calls reach.
signalling = gsm ; Due to this GSM card, we define it as GSM signal.
;pin=1234 ; Set up PIN number, no need to enable if no definition.
channel => 1 ; Channel number setting, it represents different span.
context = default ;N/A
group = 63 ;N/A
```

### 1. Configuring dialplan

Dialplan settings of G400P/E are located in `/etc/asterisk/extensions.conf`. There is a sample setting for GSM cards.

```
[from-gsm-100]
exten => s,1,Dial(SIP/100)
exten => s,n,Hangup()
exten => sms,1,Verbose(${SMSSRC})
exten => sms,n,Verbose(${SMSTXT})
exten => sms,n,ForwardSMS(1,135xxxxxxx)
```

As we can see, the context name is corresponding to the step 1 of `extra-channels.conf`. If there is incoming call over channel 1, then asterisk will search context `from-gsm-100` and perform the s priorities of this context. They are dialing SIP extension 100, and then hanging up. If there is incoming sms over channel 1, it will be routed to sms priorities rather than s anymore. In this sample, the sms will be displayed on the console, including its sender. At the last priority, this sms will be resent to phone 135xxxxxxx. This priority is optional.

## Chapter 5 Functions

### 1. CLI commands of chan\_extra.

- **gsm debug span <span>** ;enable debug info of the specified span.
- **gsm intensive debug span <span>** ; enable more debug info of specified span.  
Like AT commands.
- **gsm no debug span <span>** ;disable debug info of specified span.
- **gsm send at <span>** ; send AT command to specified span.
- **gsm send pdu <span>**; send PDU code to specified span.
- **gsm send sms <span> <destination number> <sms contents>** ; send sms to outside.
- **gsm set debug file <file path>** ; save debug file to a specified file.
- **gsm set send sms coding <char coding>** ; set up sending codes of PDU.
- **gsm set send sms mode pdu** ;set up sending mode of PDU.
- **gsm set send sms mode text** ; set up sending mode of text.
- **gsm set send sms smsc <sms centre number>** ;setup centre number of sending sms.
- **gsm show debug** ; to check if debug has been enabled or not.
- **gsm show send sms coding**; to check the encoded mode of sending PDU.
- **gsm show send sms mode**; to check the sms sending mode.
- **gsm show send sms smsc**; to check the centre number of sending sms.
- **gsm show spans**; to show all the spans' states.
- **gsm show span <span>**; to check a specified span's detail info.
- **gsm show version**; to check chan\_extra's version.
- **gsm unset debug file** ; to disable output of debug file.
- **extra destroy channel <chan\_num>**; destroy specified channel.
- **extra restart**; to restart all extra channels.
- **extra show channel <chan\_num>**; to check more info of specified channel.
- **extra show channels**; to show all extra channels.
- **extra show status**; to check GSM cards' status.
- **extra set dnd<chan\_num> <on|off>**; to enable or disable specified channel's dnd feature(not support).
- **extra set hwgain<rx|tx> <chan\_num> <gain>**. Set up hardware gain to specified channel (not support).
- **extra set swgain <rx|tx> <chan\_num> <gain>**. Setup software gain to specified channel (not support).
- **extra show version**; to check DAHDI version.
- **gsm power off <span>** ; to disconnect power for specified span.
- **gsm power on <span>** ; to turn on power for specified span.
- **gsm power stat <span>**; to check specified span's status.
- **gsm rebad span <span\_number>** ; to refresh the specified span.

2. Applications and variables of chan\_extra in dialplan.
  - a. SendSMS(span, destination number, sms contents) ; to send sms.  
e.g SendSMS(1, 135xxxxxxx, "helb world")
  - b. ForwardSMS(span, destination number [,sms centre number]) ;forward sms to another mobile phone(termination).  
e.g ForwardSMS(1,135xxxxxxx)
  - c. Set(CHAR\_CODING=encoding); set up encoding mode of sending sms.  
e.g Set(CHAR\_CODING=GBK)
  - d. Set(SMSC=sms centre number); set up centre number of sms.  
e.g Set(SMSC=861380755500)
  - e. \${SMSSRC} ; sms sender number.
  - f. \${SMSTXT}; sms contents.
  - g. \${SMSPDU}; PDU codes of sms.

### 3. Sending SMS.

There are two modes for sending SMS, PDU mode and text mode. PDU mode supports a few languages' characters for sending. Text mode currently only supports english characters.

To send SMS, chan\_extra provides three methods:

- a. Sending SMS on the Asterisk console (only supports English characters).  
Usage: gsm send sms <span> <destination number> <sms contents>  
e.g gsm send sms 1 135xxxxxxx "Hello World!"
- b. Sending SMS over Linux Shell.  
Usage: asterisk -rx "gsm send sms <span> <destination number> <sms contents>"  
e.g asterisk -rx "gsm send sms 1 135xxxxxxx \"Hello World!\""
- c. Send SMS over dialplan.  
Usage: SendSMS(<span>, destination number, sms content)  
e.g SendSMS(1,135xxxxxxx, "Hello World!")

### 4. Receiving SMS.

The sms will be saved in /var/log/asterisk/sms/receive\_message when system receives. At this moment, it will trigger corresponding contexts and sms priorities over dialplan. e.g exten =>sms,1, xxxxxxxxxx(must use sms so that trigger sms receiving). It might be different for each span.

### 5. Dialing and receiving calls.

We can use SIP soft-phone to dial or receive calls. Of course, you are able to use other applications to originate calls, like Originate(). Please refer [www.voip-info.org](http://www.voip-info.org) for more.

## Chapter 6 Application Examples

### 1. Chinese SMS sending.

#### a. Sending sms in Linux Shell.

Create a shell script send\_sms.sh, and save it in ASCII encoding method (because I use GBK code). This script is like this:

```
#!/bin/sh
asterisk -rx "gsm set send sms mode pdu"
asterisk -rx "gsm set send sms coding gbk"
asterisk -rx "gsm send sms 1 135xxxxxx,/"欢迎来到中文世界/"
```

Give execute permission to it by `chmod a+x send_sms.sh`. Lastly, perform it `./send_sms.sh`.

#### b. Sending SMS in dialplan.

Increase these lines in your dialplan:

```
exten => s,1,Set(CHAR_CODING=GBK)
exten => s,1,SendSMS(1,135XXXXXXX," 欢迎来到中文世界")
```

### 2. Group sending SMS

Here is a simple sample to teach you how to send SMS in group. Firstly, prepare two files content and group. Content refers to a file that SMS contents to send. And group refers to a file to list the mobile phone numbers.

Content:

```
Hello test send sms
```

Group:

```
135xxxxxxx
136xxxxxxx
151xxxxxxx
```

Group\_sendsms.sh

```
#!/bin/sh
context=`cat ./context`
cat ./groups |
while read row; do
    asterisk -rx "gsm send sms 1 $row \"${context}\""
```

name

Give execute permission to this script and perform it by ./group\_sendsms.sh.

3. A sample to automatically save SMS to a database.

We are supposed there is a database named test, username: root, and password is test1234. This database is based on MySQL, let's take one span for an example here. In this database, there is a table named sms\_tab to save incoming SMS. And its attribute is like this:

span	smssrc(SMS sender number)	smstxt(SMS content)	time(received time)
------	---------------------------	---------------------	---------------------

Before doing this, we have to prepare some things.

- a. Configuring extra-channels.conf

```
[from-gsm-100]
; Span 1: opvxo4xx/0/1 "Open Vox G400P GSM/CDMA PCI Card 0" (MASTER) AMI/CCS
group=11
context=from-gsm-100
signalling = gsm
channel => 1
context = default
group = 63
```

- b. Creating a dialplan in extensions.conf

```
[from-gsm-100]
exten => sms,1, System(/insert_sms.sh 1 ${SMSSRC} ${SMSTXT} )
```

The system will invoke insert\_sms.sh script when executing this context.

- c. Insert\_sms.sh details:

```
#!/bin/sh
SERVER= 192.168.2.122
USER= root
PASSWORDS= test1234
DB_NAME= test
SMS_TAB= sms_tab
SPAN=$1
SMSSRC=$2
SMSTXT=$3
TIME=`date +%Y-%m-%d %H:%M:%S`
SQL= INSERT INTO $SMS_TAB VALUES ('$SPAN', '$SMSSRC', '$SMSTXT', '$TIME')
mysql -h$SERVER -u$USER -p$PASSWORDS -e"$SQL" $DB_NAME
```

Alright, at this moment the SMS will be automatically inserted to database test.

## Chapter 7 Virtual Serial

In order to operate GSM modem, OpenVox designs a virtual serial port(/dev/ttyGSMx) at the driver layer. With this method, users will be able to operate GSM modem over Minicom or implement GPRS internet access.

By means of multiplexer, DAHDI(opvxd4xx.ko) is able to handle all commands over related protocol. Therefore, users can load Asterisk as well and the virtual serial port device will be recognized as multiplexer channel.

### 1. GPRS features

There are a few steps must be donedah before using GPRS

- a. Installing dial-up tool(e.g 'yum install ppp' to install)
- b. Downloading and installing chan\_extra-2.0.5. You have to select TTY
- c. Enable tty parameter of each span definition in /etc/asterisk/extra-channels.conf. Set tty to on(default vaule is off).

```
; Span 1: opvxd4xx/0/1 "OpenVox G400P GSM/CDMA PCI Card 0" (MASTER) AMI/CCS
group=11
context=from-gsm
signalling = gsm
debugat=on
tty=on
;pin=1234
channel => 1
context = default
group = 63

; Span 2: opvxd4xx/0/2 "OpenVox G400P GSM/CDMA PCI Card 0" AMI/CCS
group=12
context=from-gsm
signalling = gsm
debugat=on
tty=on
;pin=1234
channel => 3
context = default
group = 63
```

Figure 10 Enabling tty mode

- d. Perform **dmesg** to check if ttymode has been enabled or not. Because you select TTY mode of step b. The system creates TTY nodes for each span.

```
PCI: Enabling device 0000:02:0a.0 (0000 -> 0002)
ACPI: PCI Interrupt 0000:02:0a.0[A] -> GSI 22 (level, low) -> IRQ 209
Found an OpenVox G400P: Version 1.1
card 0 opvxd4xx: slot 0 is Installed
card 0 opvxd4xx: slot 1 is Installed
card 0 opvxd4xx: slot 2 is Installed
card 0 opvxd4xx: slot 3 is Installed
opvxd4xx: card 0 slot 0 is ttymode
opvxd4xx: card 0 slot 1 is ttymode
opvxd4xx: card 0 slot 2 is ttymode
opvxd4xx: card 0 slot 3 is ttymode
```

**Figure 11 Checking tty mode information**

- e. Configuring PPP dial-up script in `../chan_extra-2.0.5/script/ppp/ppp-on`, Create and edit your own ppp-on parameters. Here is a sample of China local.

```
#!/bin/sh
NUMBER=*99*##
PDP_TYPE=IP
APN=CMNET

DEVICE=/dev/ttyGSM0
BAUD=115200

DIAL_FILE=./ppp-dial

export NUMBER
export PDP_TYPE
export APN

pppd crtscts nodetach noauth usepeerdns noipdefault ipcp-accept-remote defaultroute \
$DEVICE $BAUD connect $DIAL_FILE
```

**Figure 12 Configuring ppp dial-up script**

Please read [this](#) for more usages of pppd tool.

- f. Performing ppp-on script by `./ppp-on`. You will see it like this:

```
send (ATDT*99*#^M)
timeout set to 10 seconds
Waiting for connect...
expect (CONNECT)

^M

^M
CONNECTCONNECT
-- got it

send (^M)
Connect Successfully!

Serial connection established.
Using interface ppp0
Connect: ppp0 <--> /dev/ttyGSM0
Could not determine remote IP address: defaulting to 10.64.64.64
not replacing existing default route via 172.16.0.1
local IP address 10.142.189.105
remote IP address 10.64.64.64
primary DNS address 120.196.165.7
secondary DNS address 221.179.38.7
```

**Figure 13 GPRS successfully connection flow**

- g. Performing `ifconfig` to check network state:

```

ppp0    Link encap:Point-to-Point Protocol
        inet addr:10.142.189.105  P-t-P:10.64.64.64  Mask:255.255.255.255
        UP POINTOPOINT RUNNING NOARP MULTICAST  MTU:1500  Metric:1
        RX packets:6 errors:0 dropped:0 overruns:0 frame:0
        TX packets:8 errors:0 dropped:0 overruns:0 carrier:0
        collisions:0 txqueuelen:3
        RX bytes:72 (72.0 b)  TX bytes:130 (130.0 b)

```

**Figure 14 GPRS internet access state**

## 2. Minicom features.

In order to use minicom to operate GSM modem. You have to configure minicom terminal first. Here are the steps:

- a. Before doing this, you must finish steps b and c of GPRS features section.
- b. Executing minicom -s, and you will see this:

```

r-----[configuration]-----r
|  Filenames and paths          |
|  File transfer protocols      |
|  Serial port setup           |
|  Modem and dialing           |
|  Screen and keyboard         |
|  Save setup as dfl            |
|  Save setup as..             |
|  Exit                         |
|  Exit from Minicom           |
|-----|-----|

```

**Figure 15 Minicom-Serial port setup**

- c. Select “Serial port setup” and type [enter], and configure it like this(I use the first span to test).

```

A - Serial Device      : /dev/ttyGSM0
B - Lockfile Location  : /var/lock
C - Callin Program    :
D - Callout Program    :
E - Bps/Par/Bits      : 38400 8N1
F - Hardware Flow Control : Yes
G - Software Flow Control : No

Change which setting?

```

**Figure 16 Minicom-Setup serial device**

- d. Save and exit it.

```
-----[configuration]-----
| Filenames and paths          |
| File transfer protocols      |
| Serial port setup           |
| Modem and dialing           |
| Screen and keyboard         |
| Save setup as dfl           |
| Save setup as..             |
| Exit                         |
| Exit from Minicom           |
-----
```

Figure 17 Minicom-configuration finishes

- e. Then you will enter minicom terminal interface. And you can see it like this:

```
Welcome to minicom 2.1

OPTIONS: History Buffer, F-key Macros, Search History Buffer, I18n
Compiled on Jan 7 2007, 04:48:39.

Press CTRL-A Z for help on special keys

AT S7=45 S0=0 L1 V1 X4 &c1 E1 Q0
OK

+CSQN: 24, 0
ate1
OK
at+cmgf=1
OK
at+c
+CSQN: 23, 0
mgs="13751080347"
> hello world
>
+CSQN: 22, 0
+CMGS: 33
```

Figure 18 Minicom interface

In this illustration, I sent a sms “hello world” to a test phone. And it successfully received the sms.

- f. Checking spans status. On the Asterisk CLI, you can see it as below, on which has been used as Multiplexer mode.

```
*CLI> gsm show spans
GSM span 1: Power on, Provisioned, Up, Active, Multiplexer
GSM span 2: Power on, Provisioned, Up, Active, Multiplexer
```

**Figure 19 Multiplexer status**

The enabled spans are recognized as multiplexer channels.

- g. An example of all the spans status:

```
gsm_elastix*CLI> gsm show spans
GSM span 1: Power on, Provisioned, Up, Active, Multiplexer
GSM span 2: Power on, Provisioned, Up, Active, Multiplexer
GSM span 3: Power on, Provisioned, Undetected SIM Card, Active, Standard
GSM span 4: Power on, Provisioned, Up, Active, Standard
GSM span 5: Power on, Provisioned, Up, Active, Standard
GSM span 6: Power on, Provisioned, Up, Active, Standard
GSM span 7: Power on, Provisioned, Up, Active, Standard
GSM span 8: Power on, Provisioned, Up, Active, Standard
```

**Figure 20 Spans' status**

As you can see, there are two spans have been configured as Multiplexer. The third span has not been inserted SIM card. Others are setting as regular channels.