Sony Erricson GC82 How To

Documented by Roger
roger<at>eskimo.com
2004.01.17

Original posting of this “how to”:
http://www.eskimo.com/~roger/files/doc/howto-gc82.html

Tarball of files:
http://www.eskimo.com/~roger/files/packages/gc82-20050117.tar.bz2

Table of Contents

1) Enable serial_cs within the Linux Kernel

This is only if you are not using a pre-made kernel from a Linux distribution -- then, I would believe this has already been done for you.

```
/usr/src/linux # make menuconfig
```

Device Drivers > Character Devices > Serial Drivers >

```
<*> 8250/16550 and compatible serial support
[ ]   Console on 8250/16550 and compatible serial port
<M>   8250/16550 PCMCIA device support
[ ]   8250/16550 device discovery via ACPI namespace
(4)   Maximum number of non-legacy 8250/16550 serial ports
[ ]   Extended 8250/16550 serial driver options
--- Non-8250 serial port support
```

And now, as usual, rebuild & reboot the new kernel.

Note: I compiled the serial_cs as module and only found one serial port (ie. /dev/ttyS2) being linked upon inserting the card. With the serial_cs as module, I'm finding two serial ports being linked when the card is inserted (ie. /dev/ttyS0 & /dev/ttyS2).

2) Hack the ppp auth source code

There are two possible patches on the web and both work fine against net-dialup/ppp-2.4.1-r14 (gentoo).

(See step #3 before proceeding with this Step #2 as Step #2 is not required when providing the ":10.0.0.1" option within ppp peers.)

Place one of the two patches listed below within the net-dialup/ppp/files folder. Add one of the following lines to the ppp-2.4.1-r14.ebuild and uncomment the one for the patch you're using:\

```
$ cat /usr/local/portage/net-dialup/ppp/ppp-2.4.1-r14.ebuild  |grep auth.c
        #epatch ${FILESDIR}/auth.c.segc82.patch
        epatch  ${FILESDIR}/auth.c.segc82.patch.new
```

You may have to do a "ebuild /usr/local/portage/net-dialup/ppp/ppp-2.4.1-r14.ebuild digest", but just "emerge /usr/local/portage/net-dialup/ppp/ppp-2.4.1-r14.ebuild" should work just fine & dandy. I'm using PORTDIR_OVERLAY=/usr/local/portage in make.conf to preserve my work, otherwise, you can just use your /usr/portage folder but beware this work with be gone on a emerge sync.

```
$ cat /usr/local/portage/net-dialup/ppp/files/auth.c.segc82.patch
--- ppp-2.4.1.pppoe4.orig/pppd/auth.c   2004-06-22 03:02:52.566353000 -0400
+++ ppp-2.4.1.pppoe4/pppd/auth.c        2004-06-22 03:07:41.003504064 -0400
@@ -1759,6 +1759,11 @@
     u_int32_t addr;
 {
     addr = ntohl(addr);
+/*  Hook for Sony Ericsson Edge PC Card GC82 to let 127.0.0.2 through */
+       if (2130706434 == addr) {
+               fprintf(stderr, "SE GC82 Hack -- Permitting 127.0.0.2\n");
+               return 0;
+       }
     return (addr >> IN_CLASSA_NSHIFT) == IN_LOOPBACKNET
        || IN_MULTICAST(addr) || IN_BADCLASS(addr);
 }

$ cat /usr/local/portage/net-dialup/ppp/files/auth.c.segc82.patch.new
--- pppd.orig/auth.c    2004-06-22 13:47:40.870430000 -0400
+++ pppd/auth.c 2004-06-22 13:52:03.552496592 -0400
@@ -1758,9 +1758,23 @@
 bad_ip_adrs(addr)
     u_int32_t addr;
 {
-    addr = ntohl(addr);
-    return (addr >> IN_CLASSA_NSHIFT) == IN_LOOPBACKNET
-       || IN_MULTICAST(addr) || IN_BADCLASS(addr);
+// GC82 hack to allow 127.0.0.2
+if((addr>>IN_CLASSA_NSHIFT) == IN_LOOPBACKNET) {
+  printf("class a octet of addr in loopback network\n");
+  printf("forcing exit with success anyway!!!!!!\n");
+  return 0;
+} else if(IN_MULTICAST(addr)) {
+  printf("addr is in multicast network\n");
+  return 1;
+} else if(IN_BADCLASS(addr)) {
+  printf("addr is in a bad class\n");
+  return 1;
+} else {
+  return 0;
+}
+//    addr = ntohl(addr);
+//    return (addr >> IN_CLASSA_NSHIFT) == IN_LOOPBACKNET
+//     || IN_MULTICAST(addr) || IN_BADCLASS(addr);
 }

 /*
```

3) Create ppp config files

Note, ":10.0.0.1" is needed to avoid being assigned the 127.0.0.2 address for routing. The AT&T network assigns your box 127.0.0.2 for the default gateway in order to route packets through the windows ndiswan device drivers in windows. (I'm wondering if a Linux WAN driver could be utilized to do what the Windows ndiswan driver does on windows – which I'm guessing the optimizes the tcp/ip headers or some other optimization?) Since we are not using windows (winxp), and apparently there are only drivers for the MAC O/S and Windows, we will have to bypass the optimization hack they use and go with communicating directly with the AT&T network using ":10.0.0.1".

```
$ cat /etc/ppp/peers/attws
# This optionfile was generated by pppconfig 2.1.
#
#
hide-password
noauth
#-detach
connect "/usr/sbin/chat -v -V -t3 -f /etc/chatscripts/attws"
#disconnect "/usr/sbin/chat -v -f /etc/chatscripts/attws-disconnect"
#debug
/dev/ttyS2
#230400
115200
defaultroute
noipdefault
usepeerdns
#local
remotename attws
ipparam attws
#usepeerdns
#nodetach
crtscts
#lock
#passive
novj
novjccomp
#lcp-echo-failure 4
#lcp-echo-interval 65535
#user " "

#recieve-all
#nopcomp
#noacomp
#nomagic
#modem
#noccp

nodeflate
:10.0.0.1
nobsdcomp


$ cat /etc/chatscripts/attws
# This chatfile was generated by pppconfig 2.1.
# Please do not delete any of the comments.  Pppconfig needs them.
#
# ispauth CHAP
# abortstring
#ABORT BUSY ABORT 'NO CARRIER' ABORT VOICE ABORT 'NO DIALTONE' ABORT 'NO DIAL TONE' ABORT 'NO ANSWER' ABORT DELAYED
#TIMEOUT 10
#ABORT 'BUSY' ABORT 'NO ANSWER' ABORT 'NO CARRIER'
#SAY 'Starting GPRS connect script\n'

# modeminit
#'' ATZ
# Eo=No echo, V1=English result codes
#OK     'ATE0V1'
'' AT+cfun=1
OK AT+cgreg=1
OK AT
#TIMEOUT 40

#Try some optional AT Commands
# /begin att
OK AT&F&D2&C1E0V1S0=0
OK AT+IFC=2,2
OK ATS0=0
OK AT
OK AT&F&D2&C1E0V1S0=0
OK AT+IFC=2,2
# /end att
#AT&FE0S0=0
#AT&F0&D2+IFC=2,2V1Q0XIS0=0S7=50+CMEE=1

# Set Access Point Name (APN)
SAY     'Setting APN\n'
#REG:\s1 AT+cgdcont=1,"IP","proxy"
OK      'AT+CGDCONT=1,"IP","proxy"'
# ispnumber
SAY     'Dialing...\n'
#REG:\s1 'ATD*99***1#'
OK      ATD*99***1#
#OK     'ATD*##***##'
OK

# ispconnect
#CONNECT \d\c
CONNECT ''

#OK-AT-OK ATDTreplace_with_number
# ispconnect
#CONNECT \d\c
#prelogin
#ispname''
#isppassword''
#postlogin

# end of pppconfig stuff




# cat /etc/chatscripts/attws-disconnect
#
""      "\K"
""      "+++ATH0"
SAY     "GPRS disconnected"

# cat /etc/ppp/pap-secrets
(none) * password
"user" MyPeer "passwd"
" " attws "dummy"
"Sierra Wireless" ac750
#"Sierra Wireless" ac7x0 ""
```

(As you can see by all the "#", I have done quite a bit of bit-banging. ;-)

*Note: Different radio towers or service providers may require slighly augmented chat scripts! I noticed while roaming, "novj" & "novjccomp" is required as noted in the above ppp options posted including the chat script posted above. But if you plan on using the standard AT&T network, you will need to modify the script file as posted below.

I also noticed that some servers (or service providers), respond differently with either a "CONNECT" or "OK" after the dial command. Sometimes I noticed that either or were accepted.

```
$ cat /etc/chatscripts/attws.att

# This chatfile was generated by pppconfig 2.1.
# Please do not delete any of the comments.  Pppconfig needs them.
#
# ispauth CHAP
# abortstring
#ABORT BUSY ABORT 'NO CARRIER' ABORT VOICE ABORT 'NO DIALTONE' ABORT 'NO DIAL TONE' ABORT 'NO ANSWER' ABORT DELAYED
#TIMEOUT 10
#ABORT 'BUSY' ABORT 'NO ANSWER' ABORT 'NO CARRIER'
#SAY 'Starting GPRS connect script\n'

# modeminit
#'' ATZ
# Eo=No echo, V1=English result codes
#OK     'ATE0V1'
'' AT+cfun=1
OK AT+cgreg=1
OK AT
TIMEOUT 40

#Try some optional AT Commands
# /begin att
#OK AT&F&D2&C1E0V1S0=0
#OK AT+IFC=2,2
#OK ATS0=0
#OK AT
#OK AT&F&D2&C1E0V1S0=0
#OK AT+IFC=2,2
# /end att
#AT&FE0S0=0
#AT&F0&D2+IFC=2,2V1Q0XIS0=0S7=50+CMEE=1
# Set Access Point Name (APN)
SAY     'Setting APN\n'
REG:\s1 AT+cgdcont=1,"IP","proxy"
#OK     'AT+CGDCONT=1,"IP","proxy"'
# ispnumber
SAY     'Dialing...\n'
#REG:\s1 'ATD*99***1#'
OK      ATD*99***1#
#OK     'ATD*##***##'
#OK

# ispconnect
#CONNECT \d\c
CONNECT ''

#OK-AT-OK ATDTreplace_with_number
# ispconnect
#CONNECT \d\c
#prelogin
#ispname''
#isppassword''
#postlogin

# end of pppconfig stuff
```

4) Connect to the network

I differentiated the file names for the chat and ppp options files by appending ".att" & ".roaming".

So now, I do either of the following in order to activate a wireless connection with the network:

```
$ pon attws.att
```

or

```
$ pon attws.roaming
```

(The attws-disconnect chat script isn't really needed from what I see. Matter of fact, the GC82 is particular about being initialized and the disconnect chat script seems to just make things messier for me.)

5) User space utilities

The first step you must perform before using the GC82 via a user space utility that supports the AT Command set is to initialize the GC82. The GC82 will accept no other AT Command until you send the GC82 the “AT+cfun=1”. This, I believe allows the GC82 to accept AT Commands.

```
# /usr/sbin/chat -t2 '' AT+cfun=1 > /dev/ttyS2
```

Chat is apart of the net-dialup/ppp package.

You may need to send the cfun=1 AT Command twice or you may receive an error.

Gsmlib

net-misc/gsmlib

http://www.pxh.de/fs/gsmlib/

I'm using net-misc/gsmlib-1.11_pre030826 and find gsmlib works with the GC82.

List all features (Will hang halfway on an unsupported feature. I'm guessing the FORW option is the problem.)

```
$ gsmctl -d /dev/ttyS2 all
```

Show the signal strength. After the modem is first successfullly initialized with the cfun=1 AT Command, the signal will show “99” for ~5 seconds depending on signal quality. After the ~5 seconds, you will then show the actual signal level (ie. 13). Remember, this device uses radio signals and if the signal strength is too low, you will also receive errors with gsmctl commands! The siganl strength command follows:

```
$ gsmctl -d /dev/ttyS2 sig
```

List available operators.

```
$ gsmctl -d /dev/ttyS2 op
```

You can do other operations with gsmctl such as register & deregister using an operator, set forwarding, and set locks & pins. Read 'man gsmctl' for more information.

Send text sms messages easily from the command line!

```
$ gsmsendsms -d /dev/ttyS2 6195040000 "Hello, World" -r
```

There are also daemon tools (ie gsmsmsstore and gsmsmsd) to monitor for sms messages, somewhat like setting up postfix and will deliver messages to folders such as “sent”, “failed” and “spool”. However, using this utility locks the GC82, preventing ppp from using the device (or vice versa).

Gnokii

net-dialup/gnokii

http://www.gnokii.org/

I'm guessing gnokii will more then likely work with this device as long as somebody can find the correct model number to use within the gnokii.conf file. Users wishing to achieve this will need to post to the gnokii mailing list to have the developers supply you with a proper model number to use or to further troubleshoot. I find Gnokii developers more then happy to work with others to find further supported devices. For the time being, I'm more then happy with using gsmlib.

I'm guessing the Windows O/S software packaged with the GC82 seems to use it's own ppp dialer daemon in order to be able to perform simaltaneous operations with this device such as monitoring signal strength and current operator while maintaining a ppp connection. As to how to accomplish the same method on Linux more easily might be to code in an optional kernel driver to put this information into /proc which is coded just underneath the serial_cs driver. Other devices that would be able to use this optional driver would also include mobile/cell phones which also use the signal and operator functions. And to note, mobile phones can also be attached via USB and IRDA.

My Script

I'm trying to create a script utilizing the gsmlib & ppp tools in order to connect to define whether I am on a local or roaming network. Once decided, connect to the local network if possible, else use the roaming network. Currently, I do everything manually from finding the network operators & signal strength to connecting with pon. If I get something working, I will post it. But for now, I'm running into a bug with scripting “pon”, gsmlib, chat – altogether. Seems to be locking up the device completely and requires manually removing and re-inserting the GC82 card. And another note about stability, is the GC82 seems to lock up due to heat while on a roaming network. However, further debugging shows the GC82 stable after 12 hours of being located in a ~10F degrees hotter area while being connected to the local AT&T network.

You can find my gc82con.sh & gc82sig.sh scripts within the files/scripts folder.

I have worked out most of the bugs with gc82con.sh and gc82sig.sh is a simple script to check for signal strength while moving around your antenna.

The notes on how to use the gc82con.sh are within the top of the document. The script relies on my /etc/chatscripts/attws.att & /etc/chatscripts/attws.roaming chatscripts modified according to this GC82 Howto. Do not forget to modify the $DEVICE variable to point towards your tty device. If you have any ideas on simplifying the integration of the gc82con.sh script for other users, please feal free to email your suggestions. Creating one simple chatscript and peers file would be greatly appreciated!

6) External Antennas

The external antenna adapter on the Sony & Ericsson GC82 is a propitiatory one for the time being. You can easily acquire a more steady signal by using a simple patch cord antenna or a simple magnetic vehicle mount antenna.

Both solutions are available with the GC82 propitiatory connector from Mertie and are represented in the US by Mobile Mounting Solutions, Inc. 972-838-2420 (an 800 number is available, but I do not have it handy at the moment). http://www.mobilemounts.com/

ME MTCS-8815 Patch Antenna $20.00

ME MTCS-102 Mini Mag Mount Antenna $20.00

These antenna have ~3 feet of antenna wire which is quite short but necessary when understanding the dynamics of signal strength ratio to antenna wire usage. The “patch antenna” is a small black plastic dome which can be tossed out a window, hung ~3 feet above you or simply placed around obstacles to acquire the more steady signal strength. The “mini mag mount antenna” is a small stubby antenna with a magnet at the base. I use this vehicle antenna ontop of my Ford Escape and just barely touches my Yakima Load Warrior rack to provide an even better signal.

There are other solutions such as amplifiers and other antenna models. Please feel free to email your experiences with amplifiers or other solutions. My only grudge against using an amplifier is the amplifiers will generally require an additional 12V power supply and the amplifiers will generally cost ~$200.00. Please let me know if you find the amplifiers worthwhile!

7) Final notes and rants

AT&T Data support was not too helpful but they gladly routed me to a supervisor when asked. From what I see, AT&T isn't too Linux friendly. Slightly more expensive monthly but is EDGE enabled. I currently see ~5.6K-20.0KB max. ~5.6K-12.0KB while moving on the freeway with some drop-outs and I'm doing even better using a simple vehicle mount antenna – providing me with no drop-outs until I'm out of network range and I do not need to sit still to acquire the higher speed rates. Sitting still and at a good spot on the freeway is required for the better speed rates -- preferably at a Rest Area! (As for speed rates in Linux, I'm getting about the same - both ping times and speed rates. As to the function of what ndiswan drivers do in windows, I can only guess that they slightly optimize the tcp/ip connection by reducing redundant checks. This is only a guess.)

TMOBILE Data support was glad to recognize that I used Linux and quickly routed me to a senior tech support, which only was able to provide APN and ATD info. I stated my ppp connect issues with the AC750 when connecting but he

could not assist further due to his lack of knowledge with Linux/Unix. They are slated to enable EDGE in the Summer of 2004 -- supposedly. The monthly fees are cheaper then AT&T Wireless. All in all, I dropped the AC750/TMOBILE for the faster AT&T Edge enabled option.

The data on the other websites currently is completely misleading. I've been using satellite technology for many years and I've seen nothing but inaccurate data about how satellite network devices "work in Linux" when they do not! They need additional "hacked" code to optimize tcp/ip communications -- of which, from what I understand, this "hacked" code is not GNU friendly as this "hacked" code has been used for ~5 years now.

In other words, I'm seeing people post to say it "works in Linux", only for the consumer to buy into the product and then finding it only works, but will not communicate with the servers at a desirable speed rate. Since you cannot attain the desirable speed rate without these optimized drivers/code, the device is basically dis-functional at that slow speed rate. (Reader, you can research posts I've made about Starband and usage of the "bcp/ip" protocol using Linux, you will see the same scenario displayed.)

Feel free to email me at roger<at>eskimo.com if you have any questions. Hopefully the data here will prevent people from going through the same heck as I've been through (and, all this hardware costs ~$1,000+ ;-).

8) Additional Information and Credits

I would like to credit the following sites for providing additional info:

"Linux Unwired" O'Reilly Books

http://www.oreilly.com/catalog/lnxunwired/

GC82 - AT&T Wireless Forum Thread (provides a quick howto)

http://developerboards.attwireless.com/attwsddp/board/message?board.id=techsupport&message.id=952

GC75 Linux How To

http://www.internecik.com/?p1=howto&a1=se_gc75linux

Brian at “Mobile Mounting Solutions, Inc.” for providing a patch and magnetic mount antennas and providing further information.

Phone 972-838-2420

Fax 972-838-2420

http://www.mobilemounts.com/meritie%20product%2040408.pdf

GC82 - MAC (adds some additional info)

http://www.jiwire.com/sonyericsson-gc82-edge-card-tips-troubleshooting.htm

AC750 - Manufacturers Instructions

http://www.sierrawireless.com/SupportDownload/ac7x0_linux.asp

AC750 - T-Mobile Wireless on Linux (AC750)

http://www.xmission.com/~bmidgley/tmobile/