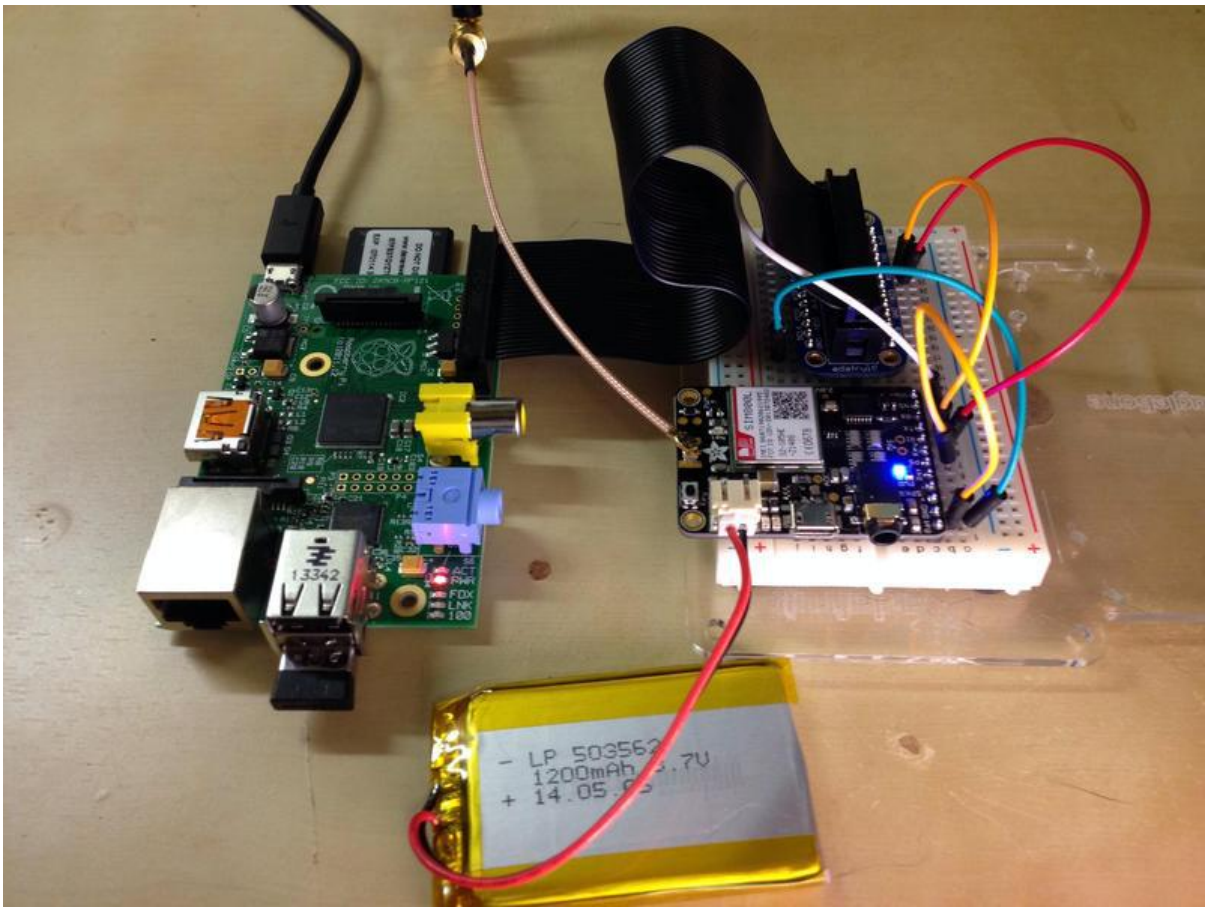
FONA Tethering to Raspberry Pi or BeagleBone Black

Created by Tony DiCola



https://learn.adafruit.com/fona-tethering-to-raspberry-pi-or-beaglebone-black

Last updated on 2023-08-29 02:35:00 PM EDT

Table of Contents

Overview

Do you need the freedom to access the internet from almost anywhere with your Raspberry Pi or BeagleBone Black? Perhaps you're building the next great internet of things hack but need that crucial link to the internet, what do you do? Why not use the same cellular phone data network that smartphones use to access the internet! With Adafruit's FONA (http://adafru.it/1946) you can tether your Raspberry Pi or BeagleBone Black to the internet through a GPRS cellular data connection and access the internet from anywhere with cell phone reception!

Terminology

First some quick definitions so you can be familiar with the terminology in this guide:

- GSM () or 'Global System for Mobile Communications' is a standard for second generation or 2G cellular phone networks. The GSM standard replaced the 1st generation analog cell phone networks a number of years ago (remember those classic 'brick' analog cell phones ()?). Although GSM is now an old standard (most smartphones you buy today support 4th generation 4G or LTE networks), GSM networks are still prevalent and available almost anywhere around the world.
- GPRS () or 'General Packet Radio Service' is a standard for sending data over a GSM cellular network. FONA supports a 2G GPRS data connection such as that offered by the T-Mobile network in the United States (unfortunately AT&T doesn't support 2G GPRS anymore).
- PPP () or 'Point-to-Point Protocol' is an old standard for connecting a computer to a network through a serial link. You might have used PPP to connect your computer to the internet through a serial modem or DSL connection years ago! With FONA a PPP connection can be created to talk to the GPRS network through FONA's serial connection with your hardware.
- APN () or 'Access Point Name' is a name to identify the gateway between a GPRS network and the internet. You will need to find the APN for the cellular network you're using with FONA. Check the cell provider's website, material that came with the SIM card, or even call the provider's support line to get the APN value.

Requirements

Next make sure you have the following before following this guide:

- FONA (http://adafru.it/1946) and supporting hardware including antenna and battery.

- Raspberry Pi or BeagleBone Black
    - NOTE: Make sure your Pi is running the Raspbian () or Occidentalis () operating system. For the BeagleBone Black you must be running the Debian () or Ubuntu image.
- SIM card with 2G GSM service including access to your cell provider's 2G GPRS data network. Access to data is typically an extra cost on top of your cell provider's voice service. Also be aware that your provider must support a 2G GPRS network--some providers only support 3G EDGE or faster connections, when in doubt ask the provider!
- APN used by your cell provider's data network. Ask your provider for this value if it wasn't given to you. Typically the APN is a short string like 'internet'. For the cell network I'm using the APN is 'web.omwtoday.com' ().
- Breadboard and hookup wires to connect the FONA to your hardware.

Be sure to read the FONA guide () for an overview of the assembly, connections, and usage of FONA before getting started.

Be very aware of the costs and policies involved with accessing your cellular provider's data network! Some providers charge very high fees at the kilobyte or megabyte level which can quickly add up to expensive phone bills! Be absolutely sure you won't incur expensive roaming data charges. Also be aware that not all providers allow tethering the data connection to a computer, and you might be kicked off their network. When in doubt ask your provider what is and isn't allowed!

Wiring

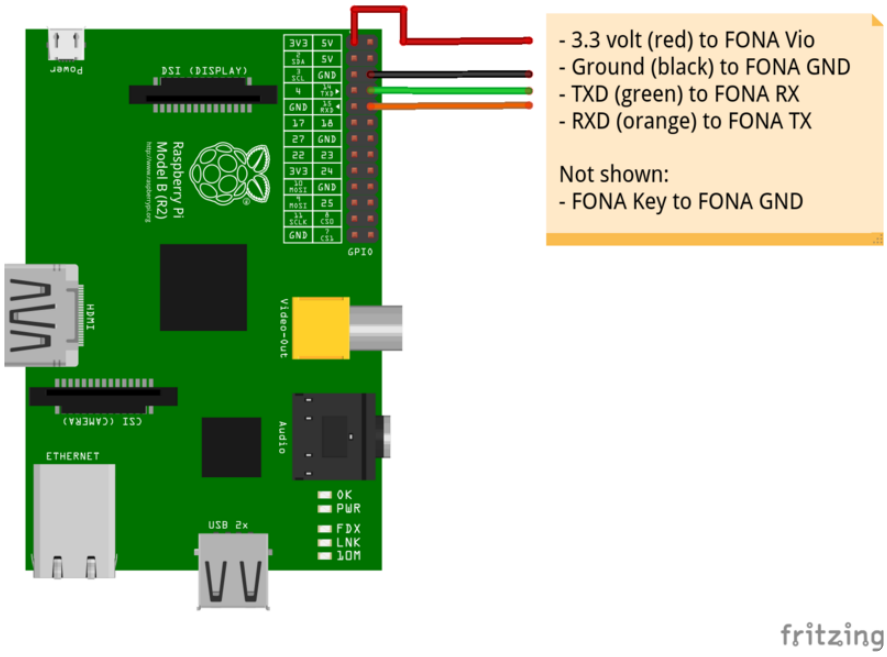
Connect FONA to your Raspberry Pi or BeagleBone Black by following the appropriate steps below.

Raspberry Pi

Wire FONA to the Raspberry Pi as follows. If you aren't familiar with the pinout of the Raspberry Pi, see this page for an image of the header pins and their functionality ().

- FONA GND to Raspberry Pi ground.
- FONA Vio to Raspberry Pi 3.3 volt power.
- FONA GND to FONA Key.
    - Note that this wiring will force FONA to stay active all the time the Raspberry Pi has power.

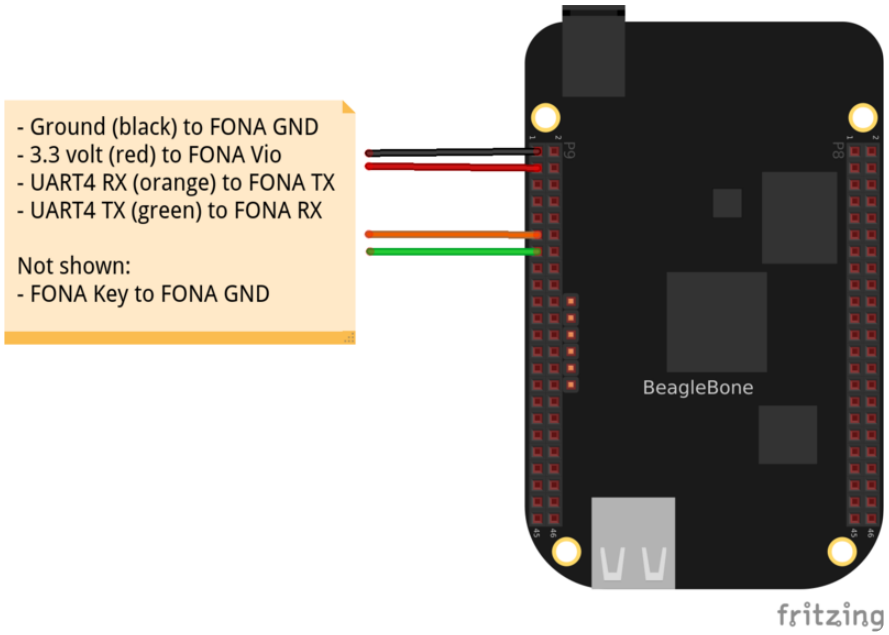
- FONA TX to Raspberry Pi RXD.
- FONA RX to Raspberry Pi TXD.



BeagleBone Black

Wire FONA to the BeagleBone Black as follows. If you aren't familiar with the pinout of the BeagleBone Black, see the Headers section of this page () for images of the pins.

- FONA GND to BeagleBone Black ground.
- FONA Vio to BeagleBone Black 3.3 volt power VDD_3V3 pin.
- FONA GND to FONA Key.
    - Note that this wiring will force FONA to stay active all the time the BeagleBone Black has power.
- FONA TX to BeagleBone Black UART4 RX pin P9_13.
    - If you aren't familiar with pin numbers on the BeagleBone Black, read the GPIO guide ().
- FONA RX to BeagleBone Black UART4 TX pin P9_11.



Setup

First you will need to do a little setup specific to your development board. Follow the steps below that are for your specific board. Note: You will need to have your board connected to the internet for the following steps so software and configuration can be downloaded.

Raspberry Pi

On the Raspberry Pi you will need to disable the kernel's use of the hardware serial connection. By default when the Raspberry Pi boots it will use the serial connection to print messages from the kernel which will confuse FONA. However it's easy to disable this serial usage by running Andrew Scheller's handy serial console script (). Connect to your Raspberry Pi's command terminal and execute:

```
sudo wget https://raw.github.com/lurch/rpi-serial-console/master/rpi-serial-console -O /usr/bin/rpi-serial-console &amp;&amp; sudo chmod +x /usr/bin/rpi-serial-console
sudo rpi-serial-console disable
```

Once the commands above are run, reboot your Raspberry Pi and connect again to the command terminal.

For reference, when you're done using FONA and wish to enable the kernel serial console again you can run the script with:

```
sudo rpi-serial-console enable
```

BeagleBone Black

On the BeagleBone Black you'll need to enable a serial UART in the device tree. By default the BeagleBone Black uses its serial UART ports for GPIO and other uses. However by loading a device tree overlay () you can enable a serial UART.

The BeagleBone Black supports up to 4 serial UARTs (). In this guide I found it's easiest to use UART4 and will describe its usage below.

The best way to enable UART4 is to modify the uEnv.txt file so the UART4 device tree overlay is loaded automatically at boot. With the BeagleBone Black connected to your computer through its USB mini connection, open the 'boot' USB storage device. Edit the file uEnv.txt on this device and add the following line at the bottom on a new line:

capemgr.enable_partno=BB-UART4

NOTE: Don't edit uEnv.txt from a Windows machine as it can change line endings and cause the BeagleBone Black to require an operating system reinstall! Instead connect to the BeagleBone Black's command terminal and edit uEnv.txt by following the steps at the end of the exporting an overlay guide page ().

Once uEnv.txt has been updated, reboot your BeagleBone Black and verify you can see a /dev/ttyO4 (that's an upper case O, not a zero) device.

Software Setup

On both the Raspberry Pi and BeagleBone Black perform the steps below to install and setup the PPP software ().

First make sure software is installed by executing the following in a command terminal:

```
sudo apt-get update
sudo apt-get install ppp screen elinks
```

You can ignore any warnings about software already being installed.

Once the commands above are executed you can test communication with FONA. The screen tool will be used to open the serial connection to FONA and verify FONA responds to commands. Make sure FONA is wired to your device and turned on (the blue LED is solidly lit and the red LED periodically flashes). Then on a Raspberry Pi execute:

```
screen /dev/ttyAMA0 115200
```

Or on a BeagleBone Black execute:

```
screen /dev/ttyO4 115200
```

Screen will open the serial port and should show a blank screen. If you see a connection error double check FONA is wired correctly to your board and turned on, then try again.

Now type AT and press enter (you might not see the AT characters echoed back as you type, don't worry that's ok). You should see FONA respond with OK. If you see the OK response then communication with FONA is working great and you can close screen. To close screen press Ctrl-A and type :quit and press enter.

If you did not see the OK response, double check FONA is wired correctly to your serial port and is turned on (solid blue LED lit, periodic red flash), then try again. Don't move on until you've confirmed you can connect to FONA and get a response from it!

PPP Configuration

Next you can configure PPP by following the steps below to create a new PPP peer configuration. The steps below were adapted from Ubuntu's ADSL PPPoE guide manual configuration by hand () section.

In a command terminal execute the following commands to login as root and navigate to the /etc/ppp/peers/ directory:

```
sudo -i
cd /etc/ppp/peers/
```

Inside the peers directory there are configuration files which define how each PPP connection is setup. Download an example FONA configuration file called fona by executing:

```
wget https://raw.githubusercontent.com/adafruit/FONA_PPP/master/fona
```

Now open the file fona in a text editor by executing:

```
nano fona
```

You should see a file that looks like the following:

```
# Example PPPD configuration for FONA GPRS connection on Debian/Ubuntu.

# MUST CHANGE: Change the -T parameter value **** to your network's APN value.
# For example if your APN is 'internet' (without quotes), the line would look like:
# connect "/usr/sbin/chat -v -f /etc/chatscripts/gprs -T internet"
connect "/usr/sbin/chat -v -f /etc/chatscripts/gprs -T ****"

# MUST CHANGE: Uncomment the appropriate serial device for your platform below.
# For Raspberry Pi use /dev/ttyAMA0 by uncommenting the line below:
#/dev/ttyAMA0
# For BeagleBone Black use /dev/ttyO4 by uncommenting the line below:
#/dev/ttyO4

# Speed of the serial line.
115200

# Assumes that your IP address is allocated dynamically by the ISP.
noipdefault

# Try to get the name server addresses from the ISP.
usepeerdns

# Use this connection as the default route to the internet.
defaultroute

# Makes PPPD "dial again" when the connection is lost.
persist

# Do not ask the remote to authenticate.
noauth

# No hardware flow control on the serial link with FONA
nocrtscts

# No modem control lines with FONA.
local
```

This configuration file controls the options that will be set by PPPD when the FONA PPP connection is created. You can find a description of all the options in the PPPD man page () if you're curious.

Notice the two sections at the top that say MUST CHANGE. The first section controls which script is called by PPPD to tell FONA to set up a GPRS connection. This section uses the chat command () to send and expect certain strings when communicating with FONA.

The only thing you must change is the APN value on the connect line. You need to change the -T **** option at the end to be -T APN_value. For example if your APN is "internet" (without quotes), you would change the line to read:

```
connect "/usr/sbin/chat -v -f /etc/chatscripts/gprs -T internet"
```

The second thing you must change is the configuration of the serial port connected to FONA. Uncomment (remove the preceding #) the /dev/ttyAMA0 line if you're using a Raspberry Pi, or uncomment the /dev/ttyO4 line if you're using a BeagleBone Black.

The rest of the settings in the file do not need to be modified. Save the file (press Ctrl-O then enter in nano), then quit the editor (press Ctrl-X in nano).

You might have noticed that the connect line in the PPP peers configuration references a specific chat script. To see this configuration open the /etc/chatscripts/gprs file in a text editor to see what it contains. For example execute:

```
nano /etc/chatscripts/gprs
```

You should see a file like the following:

```
# You can use this script unmodified to connect to cellular networks.
# The APN is specified in the peers file as the argument of the -T command
# line option of chat(8).

# For details about the AT commands involved please consult the relevant
# standard: 3GPP TS 27.007 - AT command set for User Equipment (UE).
# (http://www.3gpp.org/ftp/Specs/html-info/27007.htm)

ABORT           BUSY
ABORT           VOICE
ABORT           "NO CARRIER"
ABORT           "NO DIALTONE"
ABORT           "NO DIAL TONE"
ABORT           "NO ANSWER"
ABORT           "DELAYED"
ABORT           "ERROR"

# cease if the modem is not attached to the network yet
ABORT           "+CGATT: 0"

""              AT
TIMEOUT         12
OK              ATH
OK              ATE1

# +CPIN provides the SIM card PIN
#OK             "AT+CPIN=1234"

# +CFUN may allow to configure the handset to limit operations to
# GPRS/EDGE/UMTS/etc to save power, but the arguments are not standard
# except for 1 which means "full functionality".
#OK             AT+CFUN=1

OK              AT+CGDCONT=1,"IP","\T","",0,0
OK              ATD*99#
TIMEOUT         22
CONNECT         ""
```

You probably don't need to modify this file, but it's good to be aware of its contents in case you need to customize how the GPRS connection is created. The format of the file is described in the chat man page (), but in general the structure is text to send on the serial connection, followed by whitespace, and then a result to expect to receive on the serial connection.

One thing to note is the commented line '+CPIN provides the SIM card PIN'. If your SIM card requires a PIN to unlock, uncomment this line and set the PIN value.

Save the file if you've made changes, and exit the text editor. You are now done configuring the PPP connection and can exit the sudo interactive root shell by running:

```
exit
```

You should be brought back to a command terminal on your Raspberry Pi or BeagleBone Black as the user you originally connected to the device. You're now ready to test and use the GPRS connection in the next section of the guide.

Usage

If you've followed the guide this far you should be at a point where you're ready to enable the GPRS connection. Just to recap, make sure you've connected FONA to your Raspberry Pi or BeagleBone Black, that you've verified FONA can communicate with your serial connection, and that you've installed the PPPD software and configured it. If you've missed any of these steps make sure to go back to the previous pages and complete them.

Before you test the GPRS connection, first disconnect any wired or wireless networking connections you might already have with your hardware and restart the device. This will help ensure there are no issues or confusion over accessing the PPP interface vs. other network interfaces. You might need to connect a monitor and keyboard to your hardware to access it when all the network connections are disconnected (however on a BeagleBone Black you can connect the USB connection to a computer and access it through its private network connection I found in my testing).

To bring up the FONA GPRS connection and setup a PPP connection, run the following command:

```
sudo pon fona
```

The command should exit with no response. Don't worry, behind the scenes PPPD should be setting up the connection. You can check the system log file to see what PPPD is doing and if it hit any errors. Execute the following command to show the log for PPPD:

```
cat /var/log/syslog | grep pppd
```

If the PPP connection was successfully created you should see something like this at the end of the log file:

```
Jul  8 03:47:40 raspberrypi pppd[2321]: pppd 2.4.5 started by root, uid 0
Jul  8 03:47:40 raspberrypi pppd[2321]: Serial connection established.
Jul  8 03:47:40 raspberrypi pppd[2321]: Using interface ppp0
Jul  8 03:47:40 raspberrypi pppd[2321]: Connect: ppp0 &lt;--&gt; /dev/ttyAMA0
Jul  8 03:47:41 raspberrypi pppd[2321]: PAP authentication succeeded
Jul  8 03:47:42 raspberrypi pppd[2321]: Could not determine remote IP address:
defaulting to 10.64.64.64
Jul  8 03:47:42 raspberrypi pppd[2321]: local  IP address 21.144.145.193
Jul  8 03:47:42 raspberrypi pppd[2321]: remote IP address 10.64.64.64
Jul  8 03:47:42 raspberrypi pppd[2321]: primary   DNS address 10.177.0.34
Jul  8 03:47:42 raspberrypi pppd[2321]: secondary DNS address 10.168.185.116
```

You might need to wait a few seconds for the PPP connection to be setup before checking the PPPD log.

One thing to look for is the flashing of the red LED on FONA--if the GPRS connection is established the red LED will flash quickly (about twice a second).

If you see an error in the PPPD log then there was likely a problem setting up the GPRS connection. You can examine the raw serial communication with FONA by looking at the chat command log. Execute the following to view the chat log:

```
cat /var/log/syslog | grep chat
```

The most recent lines will be at the end of the file. Look for any kind of error message that might indicate something unexpected has happened, like a PIN being required, APN being incorrect, etc. For reference here's what I see when a PPP connection is successfully setup with my cell provider:

```
Jul  8 03:47:40 raspberrypi chat[2323]: abort on (BUSY)
Jul  8 03:47:40 raspberrypi chat[2323]: abort on (VOICE)
Jul  8 03:47:40 raspberrypi chat[2323]: abort on (NO CARRIER)
Jul  8 03:47:40 raspberrypi chat[2323]: abort on (NO DIALTONE)
Jul  8 03:47:40 raspberrypi chat[2323]: abort on (NO DIAL TONE)
Jul  8 03:47:40 raspberrypi chat[2323]: abort on (NO ANSWER)
Jul  8 03:47:40 raspberrypi chat[2323]: abort on (DELAYED)
Jul  8 03:47:40 raspberrypi chat[2323]: abort on (ERROR)
Jul  8 03:47:40 raspberrypi chat[2323]: abort on (+CGATT: 0)
Jul  8 03:47:40 raspberrypi chat[2323]: send (AT^M)
Jul  8 03:47:40 raspberrypi chat[2323]: timeout set to 12 seconds
Jul  8 03:47:40 raspberrypi chat[2323]: expect (OK)
Jul  8 03:47:40 raspberrypi chat[2323]: AT^M^M
Jul  8 03:47:40 raspberrypi chat[2323]: OK
Jul  8 03:47:40 raspberrypi chat[2323]:  -- got it
Jul  8 03:47:40 raspberrypi chat[2323]: send (ATH^M)
Jul  8 03:47:40 raspberrypi chat[2323]: expect (OK)
Jul  8 03:47:40 raspberrypi chat[2323]: ^M
Jul  8 03:47:40 raspberrypi chat[2323]: ATH^M^M
Jul  8 03:47:40 raspberrypi chat[2323]: OK
Jul  8 03:47:40 raspberrypi chat[2323]:  -- got it
Jul  8 03:47:40 raspberrypi chat[2323]: send (ATE1^M)
Jul  8 03:47:40 raspberrypi chat[2323]: expect (OK)
```

```
Jul  8 03:47:40 raspberrypi chat[2323]: ^M
Jul  8 03:47:40 raspberrypi chat[2323]: ATE1^M^M
Jul  8 03:47:40 raspberrypi chat[2323]: OK
Jul  8 03:47:40 raspberrypi chat[2323]:  -- got it
Jul  8 03:47:40 raspberrypi chat[2323]: send
(AT+CGDCONT=1,"IP","web.omwtoday.com","",0,0^M)
Jul  8 03:47:40 raspberrypi chat[2323]: expect (OK)
Jul  8 03:47:40 raspberrypi chat[2323]: ^M
Jul  8 03:47:40 raspberrypi chat[2323]: AT+CGDCONT=1,"IP","web.omwtoday.com","",
0,0^M^M
Jul  8 03:47:40 raspberrypi chat[2323]: OK
Jul  8 03:47:40 raspberrypi chat[2323]:  -- got it
Jul  8 03:47:40 raspberrypi chat[2323]: send (ATD*99#^M)
Jul  8 03:47:40 raspberrypi chat[2323]: timeout set to 22 seconds
Jul  8 03:47:40 raspberrypi chat[2323]: expect (CONNECT)
Jul  8 03:47:40 raspberrypi chat[2323]: ^M
Jul  8 03:47:40 raspberrypi chat[2323]: ATD*99#^M^M
Jul  8 03:47:40 raspberrypi chat[2323]: CONNECT
Jul  8 03:47:40 raspberrypi chat[2323]:  -- got it
Jul  8 03:47:40 raspberrypi chat[2323]: send (^M)
```

You can see the log roughly mirrors the commands to send and expect in the /etc/chatscripts/gprs configuration. Notice my network's APN, 'web.omwtoday.com', is sent in the AT+CGDCONT command--you should see your APN on that line.

Unfortunately cellular providers might have specific restriction or requirements for setting up a GPRS connection so it's difficult to troubleshoot all possible failures in this guide. If you're running into errors try to search the web for details on configuring PPP with your provider.

If you don't see any errors in the PPPD or chat logs, congratulations your FONA's GPRS connection should be tethered to your Raspberry Pi/BeagleBone Black! To check that there is a new ppp network interface run the following command:

```
ifconfig
```

You should see a few interfaces, but in particular there should be a 'ppp0' interface. For example I see:

```
ppp0      Link encap:Point-to-Point Protocol
          inet addr:21.144.145.193  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:7 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:3
          RX bytes:72 (72.0 B)  TX bytes:111 (111.0 B)
```

If you don't see a ppp0 interface, carefully check the PPPD and chat logs above to make sure there isn't some error setting up the GPRS connection.

Now test pinging adafruit.com with the GPRS connection by executing:

```
ping adafruit.com
```

Make sure you disconnected all other network connections before attempting the ping command or else you might actually be sending the ping through your wired/ wireless network connection!

If the ping is successfull you should see a result like:

```
PING adafruit.com (207.58.139.247) from 21.144.145.193 ppp0: 56(84) bytes of data.
64 bytes from vps3.ladyada.net (207.58.139.247): icmp_req=1 ttl=45 time=316 ms
64 bytes from vps3.ladyada.net (207.58.139.247): icmp_req=2 ttl=45 time=275 ms
64 bytes from vps3.ladyada.net (207.58.139.247): icmp_req=3 ttl=45 time=773 ms
64 bytes from vps3.ladyada.net (207.58.139.247): icmp_req=4 ttl=45 time=293 ms
```

Press Ctrl-C to stop the ping command.

If you receive an error like 'unknown host adafruit.com' then your cell provider didn't provide a DNS configuration. Try manually adding OpenDNS servers to your resolve.conf by following these steps ().

If you receive no response with the ping command, check that your SIM and cellular provider allows you to access the internet. If possible, try using the SIM in a GSM phone with access to data to check it can access the internet.

If the ping command succeeded, now try accessing adafruit.com in a text-based web browser. Execute the following command:

```
elinks http://www.adafruit.com/
```

The elinks text-based web browser should open and load adafruit.com. Note that because the browser is text-based you will see a stripped down and minimal version of Adafruit's web site.

Notice the maximum speed of the internet connection is quite slow. Over a 2G GPRS connection you will only see about 5-10 kilobytes/second of download speed! This connection is far too slow to transfer large amounts of data, however it's perfect for talking to web services, sending emails, tweets, running a simple server, etc.

Press q and enter to close elinks.

Once you've verified you can ping and access adafruit.com then your FONA GPRS connection should be ready to use however you desire!

Remember, your cell phone provider might put limits or extra charges on your data

usage so be careful before heavily using the GPRS connection! Also be extra careful if your provider charges extra for international or roaming data access--don't be the next person to rack up a $20,000+ cell phone bill ()!

When you're done with the GPRS connection it's easy to stop it by executing:

```
sudo poff fona
```

Again no response will be written to the output, but you can confirm PPPD has stopped by checking its log file and running ifconfig to confirm the ppp0 interface is gone.

Automatic PPP Connection On Boot

If you'd like to configure your device to automatically bring up the PPP connection with FONA on boot, it's easy to do so by updating the network configuration. First make sure you've verified you can manually bring up the PPP connection in the previous steps. Also be sure you've disabled the kernel serial console on the Raspberry Pi, or updated uEnv.txt with the BB-UART4 device tree overlay on the BeagleBone Black--skipping either of those steps will fail to bring up the PPP connection on boot!

Edit the /etc/network/interfaces file by executing:

```
sudo nano /etc/network/interfaces
```

At the bottom of the file add a new interface configuration with the following text:

```
auto fona
iface fona inet ppp
    provider fona
```

This configuration will tell your device to bring up the FONA PPP peer automatically on boot. The configuration in /etc/ppp/peers/fona will be used to set up the PPP connection.

Save the file, exit the editor, and reboot your device. As the device boots up, watch the FONA LEDs to see that the red LED starts flashing quickly to indicate a GPRS connection is activated. Once the device has finished booting, log in and check the ppp0 interface is up by running ifconfig. You can also check the PPPD and chat logs with the commands from earlier in the setup. If there are errors, check the logs to see what issues might be occuring.

If there are no errors, congratulations your device is now configured to automatically

connect to the internet through FONA's GPRS connection! Enjoy accessing the internet from anywhere there is a 2G GPRS cellular connection!