La Fonera (FON2100 and FON2200)

The Fonera FON2100A is based on an Atheros system-on-a-chip (SoC). It got a MIPS 4KEc V6.4 processor. Fon's firmware for the Fonera is a derivative of OpenWrt proper, so expect no difficulties in running OpenWrt on your device.

The FON2200 is a rebuild of the FON2100. The only outward changes (apart from the label) is that it runs on 7.5V rather then 5V. Inside, the entire board was reworked. It has a simpler serial connection and gives off much less heat. They even left off the heat sink. When relevant, sections below mention where the FON2200 deviates.

Note: If you came here for instructions on how to install OpenWrt on a new, unmodified *La Fonera*, you should proceed to Enabling Telnet into RedBoot with serial access or without serial access and then to Installing OpenWrt with RedBoot.

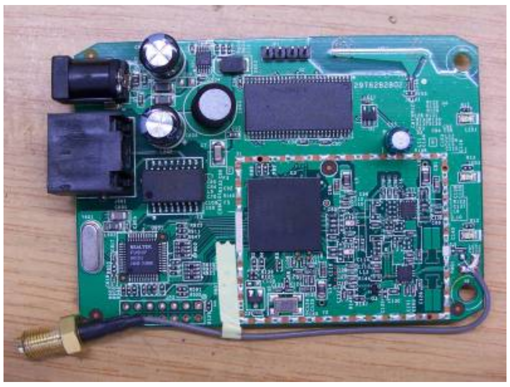
Related hardware (See additional comments below): ACCTON MR3201A, FON FON2100 A,B,C and F, Edge-Core WA3101, Philips SNR6500, SMC WEBT-G, Siemens Gigaset Wlan repeater 108

Hardware

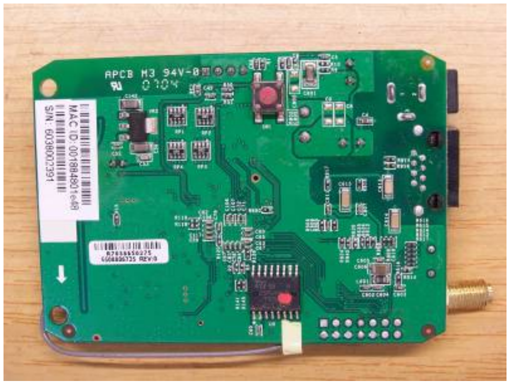
Architecture	MIPS 4KEc
Vendor	*n/a*
Bootloader	RedBoot
System-On-Chip	Atheros AR2315
CPU Speed	183 MHz
Flash size	8 MiB
RAM	16 MiB
Wireless	Integrated Atheros 802.11b/g
Ethernet	1x RJ45
USB	No
Serial	Yes
JTAG	No

- 5V power supply
- Antenna
- SPI-Bus

FON2200 PCB front



FON2200 PCB back



Power

FON2100: 5V - Internally this is regulated to 3.3V by a linear regulator, so it will probably work on anything from 4 to 6V. But higher voltage will add to the heat issues of the device. La Fonera comes with an I.T.E. Power Supply Model MU12-2050200-A1. This power supply has a 5.5mm outer diameter and 2.5mm inner diameter plug with tip-positive / ring-negative polarity. You can get a cheap replacement here [http://dx.com/p/ac-power-adapter-for-wireless-router-surveillance-security-camera-5-5x2-5mm-us-plug-100-240v-101106].

FON2200: 7.5V (small connector) - Internally this is regulated to 3.3V by a switched regulator, so it will probably work on anything from 4 to 16V, without adding to heat. A 12V supply voltage is proved to work.

voltage is proved to work.

Serial port

Warning: Theses signals are low-level (3.3V) signals NOT RS232 levels.

Note: On some models (mainly FON2100), you have to plug in the serial adapter around 3 secondes after turning the fonera on, otherwise it won't boot.

Default serial settings are **9600,8N1**

FON2100

If the ethernet jack is in front of you, it looks like this:

```
+-------------------+
|GND| . |TXD|RXD| . |
|VCC| . | . | . | . |
+-------------------+
+-----+ +--------+    +---+
|Power| |Ethernet|    |Ant|
```

Picture with labels [http://www.hidaba.com/wp-content/uploads/2007/11/fonera-serial.jpg] Another picture with labels [http://www.shadowandy.net/wp/wp-content/uploads/laFoneraSerialPinout.jpg]

Here [http://fonblog.files.wordpress.com/2006/10/fonera_serial.jpg], you can have a visual confirmation of pining for FON2100 serial port [http://fonblog.files.wordpress.com/2006/10/fonera_serial.jpg] with corresponding signals:

Signal	**Color**
VCC (3.3V)	red
GND	blue
RX	orange
TX	white

FON2200

Looking inside, with the connectors at your left hand:

```
---+  +----------------+
 P |  |GND RXD TXD 3.3V|
 O |  +----------------+
 W | +---------------------+
 E | |RAM                  |
 R | |                     |
```

Open-Mesh OM1P

Looking inside, with the connectors at your left hand:

```
---+  +----------------+
 P |  |3.3V RXD TXD GND|
 O |  +----------------+
 W | +---------------------+
 E | |RAM                  |
 R | |                     |
```

GPIO

GPIO	Description
0	TP3
1	5 of SW1
2	WLAN LED
3	pin 1 of SW1
4	pin 2 of SW1
5	RESET (!)
6	RESET button
7	pin 6 of SW1

Case

To open the case remove the two rubber feet on the opposite site to the antenna jack, they will reveal two crosspoint screws.

Additional Comments

Detailed Information may be found in the FCC database: Doing a search on Fonera's FCC-ID reveals, that this device is actually made by Accton/SMC (http://www.accton.com.tw/ [http://www.accton.com.tw/]).

A look at the varius test reports and external photos shows, that this device is provided also under the Trade(Model) Names

- ACCTON MR3201A
- FON FON2100A,B,C and F
- Edge-Core WA3101
- Philips SNR6500
- SMC WEBT-G
- Siemens Gigaset Wlan repeater 108

Copies of the mentioned FCC-documents may also be found at http://mobileaccess.de/fonera/ [http://mobileaccess.de/fonera/]

According to OpenWrt, the output-power is set to 18dBm, in contrast to the FCC-RF-tests where an output power of almost 25dBm (actually 24.89dBm on channel 6 in 802.11g mode) is measured. Question here is, if further versions will provide an adjustable TX-Power up to that level. Is the information provided by OpenWrt of 18dBm just a hypothetical value? I found no reference in Kamikaze on how to adjust it.

Another interessting issue is the possible frequency range, as specified by Atheros itself at http://www.atheros.com/pt/bulletins/AR5006AP_GBulletin.pdf [http://www.atheros.com/pt/bulletins/AR5006AP_GBulletin.pdf] - where the available band ranges from 2.300GHz to 2.500GHz. Question here is, if there will be an option to use further frequencies than the international standardized 14 Channels. In particular, there might be a very high interest e.g. from the amateur-radio community!

frequencies than the international standardized 14 Channels. In particular, there might be a very high interest e.g. from the amateur radio community.

JTAG

There seems to be a 14 pin unpopulated JTAG, but it is not that important as the RedBoot boatloader does not seem to be crippled.

Original software

Booting time output

```
+PHY ID is 0022:5521
Ethernet eth0: MAC address xx:xx:xx:xx:xx
IP: 0.0.0.0/255.255.255.255, Gateway: 0.0.0.0
Default server: 0.0.0.0
RedBoot(tm) bootstrap and debug environment [ROMRAM]
Non-certified release, version v1.3.0 - built 16:57:58, Aug  7 2006
Copyright (C) 2000, 2001, 2002, 2003, 2004 Red Hat, Inc.
Board: ap51
RAM: 0x80000000-0x81000000, [0x80040450-0x80fe1000] available
FLASH: 0xa8000000 - 0xa87f0000, 128 blocks of 0x00010000 bytes each.
== Executing boot script in 1.000 seconds - enter ^C to abort
^C
RedBoot>
```

Boot timing

```
power                t=0
safe to connect TX   t+~3sec
+                    t+7sec
PHY ID               t+9sec
redboot banner       t+13sec
WLAN light blinks    t+86sec
```

Maybe you think your fonera serial port is not working due to bogus serial pinouts posted on Hungarian blog pages. Maybe you started cursing greedy mouth-breathing corporations and ranting about the need for GPLv3. If so, the timing above may help you restore your calmness. You should see the '+' within 7 seconds of poweron. The problem "my fonera doesn't boot with serial port plugged in" stops the '+' from appearing, so if you see '+' you haven't got that problem.

Note: FON2200 seems to have more full-featured redboot, as well as address 192.168.1.1 set with a 2 second timeout. So theoratically, you could telnet directly into the Redboot, without serial or modification.

Flash layout

```
RedBoot> fis list
Name              FLASH addr  Mem addr    Length      Entry point
RedBoot           0xA8000000  0xA8000000  0x00030000  0x00000000
rootfs            0xA8030000  0xA8030000  0x00700000  0x00000000
vmlinux.bin.l7    0xA8730000  0x80041000  0x000B0000  0x80041000
FIS directory     0xA87E0000  0xA87E0000  0x0000F000  0x00000000
RedBoot config    0xA87EF000  0xA87EF000  0x00001000  0x00000000
```

As you can see, the vmlinux.bin.l7 partition is of B0000, ergo 720896 bytes length. That means, that the kernel size can not exceed 720kb without reinitializing the FIS (which is fine if you are prepared to rewrite the root FS as well). Like wise the rootfs partition has a size of 0x700000 or 7340032 bytes. You've got 7.3 MiB space.

Backing up the Original Firmware

After gaining SSH access [http://stefans.datenbruch.de/lafonera/] use these commands:

```
cd /dev/mtdblock
httpd -p 9090
```

Connect to the Fonera through the private network. Now you can download the mtd partiotions using the addresses:

```
http://192.168.10.1:9090/0ro
http://192.168.10.1:9090/1ro
http://192.168.10.1:9090/2ro
http://192.168.10.1:9090/3ro
http://192.168.10.1:9090/4ro
http://192.168.10.1:9090/5ro
http://192.168.10.1:9090/6ro
http://192.168.10.1:9090/7ro
```

Also take note of the output of the command

```
cat /proc/mtd
```

That should be:

```
dev:    size   erasesize  name
mtd0: 00030000 00010000 "RedBoot"
mtd1: 006f0000 00010000 "rootfs"
mtd2: 00560000 00010000 "rootfs1"
mtd3: 00010000 00010000 "config"
mtd4: 000b0000 00010000 "vmlinux.bin.l7"
mtd5: 0000f000 00010000 "FIS directory"
mtd6: 00001000 00010000 "RedBoot config"
mtd7: 00010000 00010000 "board_config"
```

Restoring the Original Firmware

Updating / Unbricking via RedBoot

First, download the original firmware and extract it to the machine where you run the TFTP server:

```
$ wget -q -O - http://downloads.fon.com/firmware/current/fonera_0.7.1.1.fon | tail -c +520 - | tar xvfz -
upgrade
rootfs.squashfs
kernel.lzma
hotfix
$ cp kernel.lzma /tftp/
$ cp rootfs.squashfs /tftp/
```

Replace `/tfptp/` above with the name of the directory from where you serve your TFTP files (on a standard Debian installation this would be `/srv/tftp`). Make sure your TFTP server is running and ready to serve. `atftpd` has been proven to work well.

`kernel.lzma` and `rootfs.squashfs` are the two files which will be written to the flash storage of the router.

Now access the prompt of the bootloader as described further below and proceed as if flashing OpenWrt (see below), using of course the two files mentioned above as kernel and rootfs image instead.

Fonera's bootloader: RedBoot

The Fonera routers use *RedBoot* as their boot loader. *RedBoot* allows for both serial access as well as Telnet access. An unmodified *FON 2100* does not have Telnet enabled, unfortunately, so in order to enable that, further steps have to be taken (see further below). To access the serial console, a TTL level converter has to be attached to the serial port and connected to an RS-232 port.

Enable Telnet into RedBoot, with Serial Access

If you have access to serial console of RedBoot (only possible if opening the case and attaching a TTL line converter cable), then you can enable Telnet for future convenience by changing a few RedBoot configuration variables.

The default form of the `fconfig` command will force you to enter the data, change and confirm every initialized variable. To avoid reentering the `boot_script_*` data and harming unnecessary variables, run the `fconfig list` command first to look at variable names and values:

```
RedBoot> fconfig -l -n
boot_script: true
boot_script_data:
.. fis load -l vmlinux.bin.l7
.. exec
boot_script_timeout: 1
bootp: false
bootp_my_gateway_ip: 0.0.0.0
bootp_my_ip: 0.0.0.0
bootp_my_ip_mask: 255.255.255.255
bootp_server_ip: 0.0.0.0
console_baud_rate: 9600
gdb_port: 9000
info_console_force: false
net_debug: false
```

Next, change only necessary variables (using their names from the previous listing) and update the RedBoot non-volatile configuration after the last change:

```
RedBoot> fconfig boot_script_timeout 10
boot_script_timeout: Setting to 10
Update RedBoot non-volatile configuration - continue (y/n)? n
RedBoot> fconfig bootp_my_ip 192.168.5.22
Update RedBoot non-volatile configuration - continue (y/n)? n
RedBoot> fconfig bootp_my_ip_mask 255.255.255.0
Update RedBoot non-volatile configuration - continue (y/n)? n
RedBoot> fconfig bootp_server_ip 192.168.5.2
Update RedBoot non-volatile configuration - continue (y/n)? y
... Erase from 0xa87e0000-0xa87f0000: .
... Program from 0x80ff0000-0x81000000 at 0xa87e0000: .
```

Note: *The configuration is only in the RAM until you update the RedBoot non-volatile configuration.* If you reset the device without updating, the configuration will not be changed. You can use changes without the update for temporary settings.

Verify the configuration by listing the aliases this time:

```
RedBoot> fconfig -l
Run script at boot: true
Boot script:
.. fis load -l vmlinux.bin.l7
.. exec
Boot script timeout (1000ms resolution): 10
Use BOOTP for network configuration: false
Gateway IP address: 0.0.0.0
Local IP address: 192.168.5.22
Local IP address mask: 255.255.255.0
Default server IP address: 192.168.5.2
Console baud rate: 9600
GDB connection port: 9000
Force console for special debug messages: false
Network debug at boot time: false
```

I specified a 10 second timeout here, so I have this 10 second time frame to telnet into RedBoot. If you are not able to hit the enter-key within 10 seconds after powering up, go for a larger time frame.

```
+PHY ID is 0022:5521
Ethernet eth0: MAC address xx:xx:xx:xx:xx:xx
IP: 192.168.5.22/255.255.255.0, Gateway: 0.0.0.0
Default server: 192.168.5.2
RedBoot(tm) bootstrap and debug environment [ROMRAM]
Non-certified release, version v1.3.0 - built 16:57:58, Aug  7 2006
Copyright (C) 2000, 2001, 2002, 2003, 2004 Red Hat, Inc.
Board: ap51
RAM: 0x80000000-0x81000000, [0x80040450-0x80fe1000] available
FLASH: 0xa8000000 - 0xa87f0000, 128 blocks of 0x00010000 bytes each.
== Executing boot script in 10.000 seconds - enter ^C to abort
```

Actually I had problems with my old BSD telnet on Slackware 11 to send a proper CTRL-C to the RedBoot console. I circumvented the problem with this small trick:

```
$ echo -e "\0377\0364\0377\0375\0006" > break
```

```
$ nc -vvv 192.168.5.22 9000 < break ; telnet 192.168.5.22 9000
Warning: Inverse name lookup failed for `192.168.5.22'
192.168.5.22 9000 open
== Executing boot script in 7.420 seconds - enter ^C to abort
ÿü^C
RedBoot> ÿüExiting.
Total received bytes: 82
Total sent bytes: 6
Trying 192.168.5.22...
Connected to 192.168.5.22.
Escape character is '^]'.
RedBoot>
```

I have to CTRL-C to abort *netcat*.

The CTRL-C problem seems to be caused by a disabled TELNET LINEMODE option. When you enable this option by creating a file `~/.telnetrc` with the following contents:

```
192.168.5.22 9000
        mode line
```

You can interrupt redboot with CTRL-C:

```
$ arping -qf 192.168.5.22 ; telnet 192.168.5.22 9000
WARNING: interface is ignored: Operation not permitted
Trying 192.168.5.22...
?Invalid command
Connected to 192.168.5.22.
Escape character is '^]'.
Escape character is '^]'.
== Executing boot script in 9.940 seconds - enter ^C to abort
^C
RedBoot>
```

The boot process is somehow signalled via the LEDs, first only the power LED is on, then the internet LED starts blinking, and when this internet LED is solid green, it is the right time to connect to the RedBoot console.

This is the point, where I disconnected the serial cable and closed the case. If the kernel is booting and SSH working, I do not need any debug-stuff in between. It is possible to unbrick the Fonera with this RedBoot console, as I can always reflash to a working firmware.

Enabling Telnet into RedBoot, WITHOUT Serial Access

As long as you can SSH into the running router, it is possible to overwrite the RedBoot configuration with a pre-made configuration partition, in order to enable Telnet access. This allows you to enable Telnet without ever accessing RedBoot through the serial port.

Thus, in order to follow this recipe, you need SSH access to the running Fonera and its OpenWrt derivative OS. By default, Fonera does not allow this for obvious reasons (they don't want people to tamper with the device and reinstall it). However, several ways exist to re-enable SSH access, all of which rely on various ways to inject code into a running Fonera. **Please refer to http://stefans.datenbruch.de/lafonera/ [http://stefans.datenbruch.de/lafonera/], which does have all the information you need.** Do not forget to enable SSH permanently after jail-breaking the device.

Note: If your Fonera was connected to the Internet proper in the past it will by now have auto-installed an updated firmware, and the abovementioned jail-breaking exploits will not work any longer. In this case, factory-reset the Fonera by holding down the reset button on the device for about 20 seconds, while the device is up and running. Then disconnect it from the Internet and do not give it network access until you have permanently enabled SSH access and disabled the auto-updater.

As we can see via 'dmesg' there is a mtd for the RedBoot config:

```
Creating 6 MTD partitions on "spiflash":
0x00000000-0x00030000 : "RedBoot"
0x00030000-0x00720000 : "rootfs"
0x00730000-0x007e0000 : "vmlinux.bin.l7"
0x007e0000-0x007ef000 : "FIS directory"
0x007ef000-0x007f0000 : "RedBoot config"
0x007f0000-0x00800000 : "board_config"
```

We can even dump that mtd content with

```
root@OpenWrt:~# cat /dev/mtd/4ro > /tmp/redboot_config
root@OpenWrt:~# strings /tmp/redboot_config
boot_script
boot_script_data
boot_script
fis load -l vmlinux.bin.l7
exec
boot_script_timeout
boot_script
bootp
bootp_my_gateway_ip
bootp
bootp_my_ip
bootp
bootp_my_ip_mask
bootp
bootp_server_ip
console_baud_rate
gdb_port
info_console_force
info_console_number
info_console_force
net_debug
```

It should be possible to use such a file to reflash other foneras in order to gain RedBoot access without ever opening the case. As long as someone can gain shell access to the Fonera, he could enable RedBoot telnet access to his Fonera and fiddle around with it.

This would be nice, but does not work just that easily, as the "RedBoot config" mtd partion is not writable:

```
root@OpenWrt:~# mtd write /tmp/redboot_config "RedBoot config"
```

Make this partition writable, by adding in `kernel/driver/mtd/mtdpart.c` after line 435

```
                        if (!(slave->mtd.flags & MTD_WRITEABLE)){
                        slave->mtd.flags |= MTD_WRITEABLE;
                        printk ("mtd: partition \"%s\" was read-only -- force writable -- CAMICIA HACK\n",
                                parts[i].name);
                        }
```

So you have to reflash the kernel with a kernel image that allows writing to the RedBoot config partition and then reflash that config partition in order to gain access to the RedBoot console.

Please note that they were not writeable for a reason. Writing "RedBoot config" is probably going to reset the FIS directory because it is on the same "erase sector". This is not a major problem since with RedBoot we can easily recreate them using the command "fis init" and to install OpenWrt we must do this anyway. However, if you write garbage into the RedBoot config partition by accident, it might be possible that you end up with a brick. So be careful here.

The basic steps are:

Replace the Kernel to Disable Write Protection

```
root@OpenWrt:~# cd /tmp
root@OpenWrt:~# wget http://ipkg.k1k2.de/hack/openwrt-ar531x-2.4-vmlinux-CAMICIA.lzma
root@OpenWrt:~# Connecting to ipkg.k1k2.de[85.10.200.90]:80
openwrt-ar531x-2.4-v 100% |*************************|   512 KB    00:00 ETA
root@OpenWrt:~# mtd -e vmlinux.bin.l7 write openwrt-ar531x-2.4-vmlinux-CAMICIA.lzma vmlinux.bin.l7
Unlocking vmlinux.bin.l7 ...
Erasing vmlinux.bin.l7 ...
Writing from openwrt-ar531x-2.4-vmlinux-CAMICIA.lzma to vmlinux.bin.l7 ... [w]
root@OpenWrt:~# reboot
```

Overwrite the RedBoot Configuration

```
root@OpenWrt:~# cd /tmp
root@OpenWrt:~# wget http://ipkg.k1k2.de/hack/out.hex
root@OpenWrt:~# Connecting to ipkg.k1k2.de[85.10.200.90]:80
out.hex 100% |*******************************| 4096 00:00 ETA
root@OpenWrt:~# mtd -e "RedBoot config" write out.hex "RedBoot config"
Unlocking RedBoot config ...
Erasing RedBoot config ...
Writing from out.hex to RedBoot config ... [w]
root@OpenWrt:~# reboot
```

The configuration we have just written sets an IP address of `192.168.1.254`, listens for a Telnet connection on port 9000, and waits for 10 seconds before booting.

The FIS table will likely be gone, so you will have to reflash the router now, after telnetting into it. However, the above procedure is permanent, so you will not need to repeat it ever again.

Telnet into RedBoot

```
$ telnet 192.168.1.254 9000
```

This, of course, requires that your machine has a route to 192.168.1.254. You can simply attach the Fonera directly to your machine by using the ethernet cable that came with the router (there's no need for a switch/hub) and configuring your ethernet port with an appropriate IP address in the 192.168.1.0 net.

It can be necesarry to ping the Fonera before connecting to RedBoot via telnet. Otherwise telnet access might be unreliable.

Now proceed as instructed further below by installing OpenWrt, or the original Fonera firmware.

OpenWrt

Installing OpenWrt with RedBoot

In order to install a new image from RedBoot you need to first gain access to RedBoot either via serial cable or Telnet. The *2100A* does by default *not* allow Telnet access to the boot loader, but it is possible to enable it without having to open the device and attaching a serial cable (see "Enabling Telnet into RedBoot without Serial Access" on how to do that).

You also need a machine where you can run a TFTP server. RedBoot will expect a TFTP server from where it will download the image files to be flashed onto the router. **Note:** It's also possible to use an HTTP server; this has been tested with om1p and did not work. If you have other experience please edit this page.)

Step 1: Downloading the OpenWrt Images

You have to download two files, namely the `.lzma` image of the kernel and the `.squashfs` image of the root filesystem. You can choose between versions *Kamikaze 8.09* and *Backfire 10.03*, where the former will use somewhat less memory. (*Attitude Adjustment 12.09* dropped support for devices with 16 MB or less RAM, which includes FON2100 and FON2200.) You want the port from the `atheros/` subdirectory. For example:

- http://downloads.openwrt.org/backfire/10.03.1/atheros/openwrt-atheros-vmlinux.lzma [http://downloads.openwrt.org/backfire/10.03.1/atheros/openwrt-atheros-vmlinux.lzma]
- http://downloads.openwrt.org/backfire/10.03.1/atheros/openwrt-atheros-root.squashfs [http://downloads.openwrt.org/backfire/10.03.1/atheros/openwrt-atheros-root.squashfs]

Copy `openwrt-atheros-vmlinux.lzma` and `openwrt-atheros-root.squashfs` to the machine where your TFTP server is running, into the directory from where TFTP is serving its files (on *Debian*, for example, this would be `/srv/tftp`. Use 'apt-get install atftpd' on Debian and all will be setup for you).

Step 2: Access and Configure RedBoot

Turn on the router, wait about ten seconds, then connect to RedBoot either through a serial connection or Telnet. Press Ctrl-C to interrupt the boot procedure. You will see the RedBoot prompt:

```
RedBoot> _
```

We now configure the address of the TFTP (or HTTP) server. Below, replace the two IP address placeholders with something that applies to you. The first IP in the command, "IP address of the server", is the IP address of the TFTP server that you have running that will serve the OpenWRT images to La Fonera. The second IP in the command, "IP address for the router" is only temporary, it must be a free IP address in the same subnet as the TFTP server. If you are connected to RedBoot via Telnet, set the "IP address for the router" to the IP address that you telneted to (192.168.1.254). The switch *-l* is for *local*.

```
RedBoot> ip_address -h <IP address of the server> -l <IP address for the router>/24
IP: [...], Gateway: 0.0.0.0
Default server: [...]
```

For example:

Step A: Stop network-manager on your local computer

```
# /etc/init.d/network-manager stop
```

Step B: Set up your computers interface with static ip

```
# ifconfig eth0 192.168.0.2/24
```

Step C: Set Redboot interface (-l) and tftp server (-h)

```
RedBoot> ip_address -h 192.168.0.2 -l 192.168.0.1/24
```

RedBoot is now ready to receive the image files from your machine.

Step 3: Download and Flash the Images

Now, load the kernel image via TFTP:

```
RedBoot> load -r -b %{FREEMEMLO} openwrt-atheros-vmlinux.lzma
Using default protocol (TFTP)
Raw file loaded [...], assumed entry at [...]
```

The target addresses given in the above message will vary, depending on kernel size. (**Note:*** to load via HTTP append "`-m HTTP`" to the `load` command. Not tested.)

Note: sometimes tftp need absolute path of the file. Example: *load -r -b %{FREEMEMLO} "/srv/tftp/openwrt-atheros-vmlinux.lzma"*

Next, we initialize the FIS (flash memory partition table). *This will wipe the currently installed system off the router, caveat emptor!*

```
RedBoot> fis init
```

Now it's time to flash the kernel image. The value of 0x80041000 for the -e and -r switches in the `fis create` RedBoot command below is the kernel entry point. *Do not change this value!* Especially disregard what the `load` command from above reported as the `assumed entry`. Note: that's small-L7 at the end, not the number 17.

```
RedBoot> fis create -e 0x80041000 -r 0x80041000 vmlinux.bin.l7
... Erase from 0xa8730000-0xa87e0000: ...........
... Program from 0x80041000-0x800f1000 at 0xa8730000: ...........
... Erase from 0xa87e0000-0xa87f0000: .
... Program from 0x80ff0000-0x81000000 at 0xa87e0000: .
```

Again, the addresses reported back to you will vary. The process will take a minute or two, the `fis create` command will block and not output any indication of progress. This is normal and no reason to be disturbed.

Now flash the rootfs:

```
RedBoot> load -r -b %{FREEMEMLO} openwrt-atheros-root.squashfs
Using default protocol (TFTP)

Raw file loaded 0x80041000-0x80200fff, assumed entry at 0x80041000
RedBoot> fis create rootfs
... Erase from 0xa8030000-0xa8730000: .........................................................................................
... Program from 0x80041000-0x80741000 at 0xa8030000: ....................................................................................................
... Erase from 0xa87e0000-0xa87f0000: .
... Program from 0x80ff0000-0x81000000 at 0xa87e0000: .
```

Note: append "`-m HTTP`" to the `load` command to load the image via HTTP (probably doesn't work, use tftp instead).

There is no need to give any other parameters to `fis create`, because the rootfs is longer than the kernel image, and we loaded it to `%{FREEMEMLO}` just as we did with the kernel image before. If you give the remaining free bytes on the flash as length, as we did, the rootfs will begin just after the kernel that we flashed before (`fis create` takes care of that) and occupy all remaining space on the flash device (because we calculated our *LENGTH* value that way).

Again, this process will take some time (a lot longer than flashing the kernel, as more bytes need to be written), and `fis create` will not report progress. Just leave the router doing its work and wait until the prompt returns.

Step 4: Check boot script

Finally check whether RedBoot is configured to boot the `vmlinux.bin.l7` segment which we created. Execute

```
fconfig -l -n
```

and check that the value of `boot_script_data` matches the one given at Enable Telnet into RedBoot, with Serial Access.

If the scripts do not match, update the boot config:

```
RedBoot> fconfig boot_script_data
boot_script_data:
.. fis load -l vmlinux.bin.1
.. exec
Enter script, terminate with empty line
>> fis load -l vmlinux.bin.l7
>> exec
>>
Update RedBoot non-volatile configuration - continue (y/n)? y
... Erase from 0xa87e0000-0xa87f0000: .
... Program from 0x80ff0000-0x81000000 at 0xa87e0000: .
RedBoot>
```

Step 5: Reboot the Router and enjoy OpenWrt

```
RedBoot> reset
```

will reboot the router. You can also just powercycle the device.

NOTE: If you changed RedBoot's baud rate to something different than 9600bps, revert that change unless your terminal program does auto baud detection – OpenWrt logs to its serial console with 9600bps, so having the same baud rate in RedBoot is a good idea.

If you have a serial connection to the Fonera you can now observe the OpenWrt boot sequence. Otherwise just wait for about two minutes until OpenWrt has booted, then connect to the router at `192.168.1.1` via telnet. This is the default address for a fresh OpenWrt installation. Set the password with `passwd` and log out again. Telnet will not work anymore from this point on, but SSH will.

It's time to access the web interface on `http://192.168.1.1 [http://192.168.1.1]` to complete basic configuration of your new router.

Automating the Installation with "expect"

It's possible to automate the installation process by using the Unix *expect* utility. This is probably only useful if you have a serial adapter to access RedBoot and need to reflash several Foneras in one swoop.

First you need to have the TFTP server working, then you need `cu` to access the serial port, and the `expect` tool to execute this script. Once you have the files needed in the tftp dir you can exec this script (save it to a file and exec it via `expect <filename>`).

The IP addresses in the script are only examples and will probably need to be changed for your particular scenario.

This script does not check whether `boot_script_data` in `fconfig` has the correct value (see above).

```
#!/usr/bin/expect -f
spawn cu -l /dev/ttyS0 -s 9600
#match_max 100000
set timeout 3600
# Look for passwod prompt
expect "enter ^C to abort"
send \003
# send blank line (\r) to make sure we get back to gui
expect "RedBoot>"
send -- "ip_address -h 192.168.1.13 -l 192.168.1.21/24\r"
expect "RedBoot>"
send -- "fis init\r"
expect "(y/n)? "
send -- "y\r"
expect "RedBoot>"
send -- "load -r -b %\{FREEMEMLO\} openwrt-atheros-vmlinux.lzma\r"
expect "RedBoot>"
send -- "fis create -e 0x80041000 -r 0x80041000 vmlinux.bin.l7\r"
expect "RedBoot>"
send -- "load -r -b %\{FREEMEMLO\} openwrt-atheros-root.squashfs\r"
expect "RedBoot>"
send -- "fis create -l 0x006F0000 rootfs\r"
expect "RedBoot>"
send -- "reset\r"
```

Automating the installation with fon-flash

Source code or binaries of the utility fon-flash can be obtained here [http://www.gargoyle-router.com/download.php]. Using this tool, simply connect your Fonera via Ethernet to your PC and run following command:

```
sudo ./fon-flash -i eth0 -c openwrt openwrt-atheros-root.squashfs openwrt-atheros-vmlinux.lzma
Reading image file openwrt-atheros-root.squashfs with 1966080 bytes, rounded to 0x001e0000
Reading image file openwrt-atheros-vmlinux.lzma with 917504 bytes, rounded to 0x000e0000
No packet.
No packet.
...
No packet.
No packet.
Peer MAC: 00:18:84:11:c1:84
Peer IP : 192.168.5.22
Your MAC: 00:ba:be:ca:ff:ee
Your IP : 192.168.5.0

Setting IP address
Setting IP address...
ip_addr -l 192.168.5.22/8 -h 192.168.5.0


Initializing partitions ...
fis init

loading file:
load -r -b 0x80100000 -m tftp file_1

creating flash partition (this may take some time)
fis create -f 0xa8030000 -l 0x006c0000  -e 0x00000000    rootfs

loading file:
load -r -b 0x80100000 -m tftp file_2

creating flash partition (this may take some time)
fis create -f 0xa86f0000 -l 0x000e0000  -e 0x80041000   -r 0x80041000   vmlinux.bin.l7

Setting boot_script_data...
fis load -l vmlinux.bin.l7
exec

Done. Restarting device..
```

Tested with OpenWRT Backfire 10.03.1 images.

Upgrading OpenWrt from within a running OpenWrt installation

NOTE: Updating the kernel in this fashion only works if the new kernel is not larger than the old one. Otherwise there won't be enough space in the FIS partition and you will have to reflash the entire memory (see above on how to do that).

```
cd /tmp
scp host:/path/to/openwrt-atheros-vmlinux.lzma .
scp host:/path/to/openwrt-atheros-root.squashfs .
mtd -e vmlinux.bin.l7 write openwrt-atheros-vmlinux.lzma vmlinux.bin.l7
mtd -e rootfs write openwrt-atheros-root.squashfs rootfs
reboot
```

Basic configuration

→ Basic condfiguration

The Fonera Pack (from sven-ola): FON + Mesh Network

Installation

After obtaining ssh and Redboot access, you can either use the manual install of Kamikaze described above or use sven-ola Freifunk upgrade [http://download.berlin.freifunk.net/fonera/readme.txt] with "EasyFlash" that does all the dirty job by itself!!

This distribution allow to create Fonera's Mesh Networks as described here [http://blog.freifunk.net/2007/fonera-pack-story]

```
1) Connect the Fonera to the ethernet jack. Use a cross linked cable.
2) Switch on the network card, and run update:
> sudo ifconfig eth0 up
> wget http://download.berlin.freifunk.net/fonera/ap51-flash-fonera-1.0-38
> sudo ./ap51-flash-fonera-1.0-38 eth0
  rootfs(0x006a0000) + kernel(0x00100000) + nvram(0x00000000) sums up to 0x007a0000 bytes
  Peer MAC: 00:18:84:15:53:20
  Peer IP : 192.168.1.254
  Your MAC: 00:ba:be:ca:ff:ee
  Your IP : 192.168.1.0
  Setting IP address...
  Loading rootfs...
  Sending rootfs, 4608 blocks...
  Initializing partitions...
  Rootfs partition size now 0x006b0000
  Flashing rootfs...
          Loading kernel...
  Sending kernel, 2048 blocks...
  Flashing kernel...
  Setting boot_script_data...
  Done. Restarting device...
```

Software Hacks

Disabling serial console

Easy hack to disable serial console without recompiling kernel, unpack vmlinuz.lzma, search for console= and replace with null and spaces using mfill in redboot. Also comment out /dev/ttyS0 in /etc/inittab. This is fconfig for Backfire 10.03.1 :

```
RedBoot> fconfig -l
Run script at boot: true
Boot script:
.. fis load -l vmlinux.bin.l7
.. mfill -b 0x802B3FF8 -l 4 -4 -p 0x6E756C6C
.. mfill -b 0x802B3FFc -l 6 -1 -p 0x20
.. exec
Boot script timeout (1000ms resolution): 10
Use BOOTP for network configuration: false
Gateway IP address: 0.0.0.0
Local IP address: 192.168.1.1
Local IP address mask: 255.255.255.0
Default server IP address: 192.168.1.10
Console baud rate: 9600
GDB connection port: 9000
Force console for special debug messages: false
Network debug at boot time: false
```

Hardware Hacks

Second antenna

FON2100

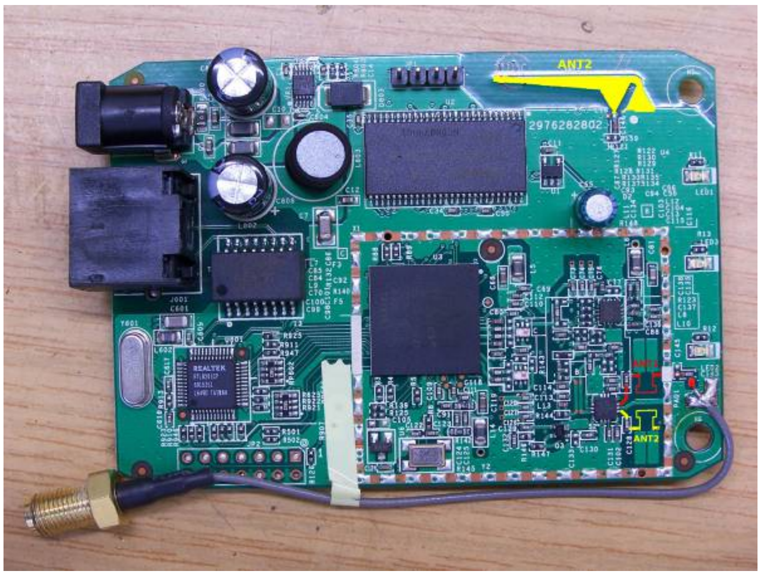
The FON2100 PCB has a pad ready to solder a second antenna. With it you can use antenna diversity to increase your signal quality in environments with reflections (e.g. due to walls) and moving clients. Check the instructions on: http://www.dd-wrt.com/wiki/index.php/LaFonera_Hardware_Second-Antenna [http://www.dd-wrt.com/wiki/index.php/LaFonera_Hardware_Second-Antenna]

Antenna]

FON2200

On the other hand FON2200 has the second antenna already etched into the PCB. Still, you can replace it with an external antenna to take further advantage of antenna diversity. Replacing the etched antenna by a regular omnidirectional external antenna like the Fonera stock one can get you more 4 dBm on antenna 2. However, this mod will only improve your link quality if antenna diversity is highly used, for instance due to moving clients. In other scenarios an unmodified FON2200 just selects antenna 1 (the best one) do to all the transmissions/receptions; adding a copy of it will improve nothing.

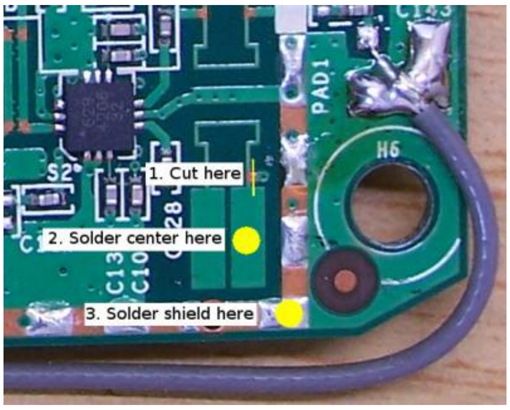
How to do it



The picture above shows an annotated photo of FON2200 PCB. On the lower right corner there are the antenna outputs, with antenna 1 highlighted in red and antenna 2 in yellow. As can be seen antenna 1 goes to the center of the coaxial cable leading to the external antenna. On the other hand antenna 2 goes to a via (through-hole path) which is then connected to ground. The etched antenna which can be seen on the upper right conner is connected to ground and thus to antenna 2 output.

In this scenario you can solder a RP-SMA pigtail connected to antenna 2 output in at least two different places.

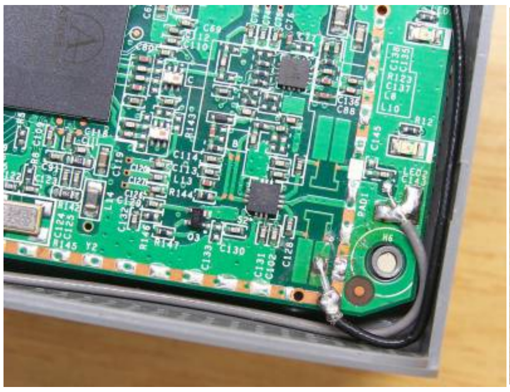
Place 1: Lower right corner, after the chip



Take a look at the picture above. You need to:

1. Cut the path which connects antenna 2 output to the via going to ground. Use a continuity tester connected between antenna 2 output and any ground point to make sure the cut was done properly.
2. Scrape out the PCB varnish coating from the middle of the right-side long rectangular pad. Solder the center of the coaxial cable from the RP-SMA pigtail here.
3. Solder the coaxial cable shield to ground. The easiest place to do so is along the rectangular copper line where FON2100 had its Faraday cage and cooler connected to ground. Along this line there are tinned points which should be used to facilitate the work.

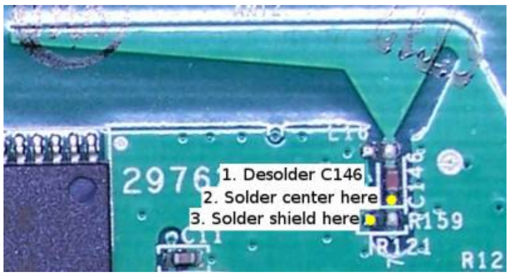
Final result should look like the pictures below.



Advantages: Tested and working, easily reversible.
Disadvantages: Difficult to solder the coaxial cable center on a point which does not come tinned from factory.

Original idea by Cardiak on Añadir una 2ª antena a fonera 2200 - Foro Seguridad Wireless [http://foro.seguridadwireless.net/hardware/anadir-una-2a-antena-a-fonera-2200/].

Place 2: upper right corner, before the etched antenna



Take a look at the picture above. You need to:

1. Desolder capacitor C146.
2. Solder the center of the coaxial cable from the RP-SMA pigtail to the inferior pad of C146.
3. Solder the shield of the coaxial cable to the left pad of the missing R159, which is connected to ground.

Advantages: Easier to solder as all points are tinned.
Disadvantages: Hardly reversible (you would have to resolder C146), not tested.

Original idea by radio3 on Fonera 2200 e 2° antenna - Forum WiFi-ITA.com [http://www.wifi-ita.com/forum/viewtopic.php?t=10451]

Hardware mp3 client

- Turning the Fonera into a hardware mp3 client [http://www.phrozen.org/fonera.html]

SPI usage: SD/MMC card reader

- Fonera SD Card Hack [http://www.larsen-b.com/Article/262.html]

SPI usage: HopeRF rfm12 433 MHz radio module

- Fonera 433 MHz rfm12 hack [http://switchsmart.org/wiki/index.php?title=La_Fonera]
- Another rfm12 hack [http://phlegmatic-prototyping.tumblr.com/post/25323114258]

1-Wire Bus

- Fonera 1-Wire Interface [http://www.karosium.com/2010/03/fonera-1-wire-microlan-extension.html]

I²C Bus

- Fonera I²C bus with GPIO [http://code.google.com/p/fonera-i2c/wiki/FoneraHacks]

Various hardware mods

- Various hw mods on dd-wrt page (32 MB RAM, second antenna, etc.) [http://www.dd-wrt.com/wiki/index.php/Category:LaFonera_Hardware_%28en%29]

Resources

- ~~Autopsy of a Fonera [http://tech.am/2006/10/06/autopsy-of-a-fonera/]~~ Dead link — *tmomas 2016/03/16 10:13*
- Get the SSH access to the Fonera [http://blog.blase16.de/index.php?url=2006/11/28/Hacking-Fonera]
- Hacking the La Fonera [http://stefans.datenbruch.de/lafonera/]
- OpenWrt development [http://forum.openwrt.org/viewtopic.php?pid=39251#p39251]
- ~~Picture of serial [http://jauzsi.hu/2006/10/13/inside-of-the-fonera] (pinout from fonera perspective: RX host→fonera)~~ Dead link — *tmomas 2016/03/16 10:14*
- Debricking and more [http://www.easy2design.de/bla/?page_id=98]
- RedBoot userguide [http://ecos.sourceware.org/docs-latest/redboot/redboot-guide.html]
- Misc Links (Italian language) [http://wiki.ninux.org/moin.cgi/La_Fonera]
- The Linux Kernel Module Programming Guide [http://www.tldp.org/LDP/lkmpg/]
- Blog about Fonera [http://karman.homelinux.net/blog/] (Spanish)
- Blog about Hacking the 0.7.1r2 firmware [http://mrmuh.blogspot.com/2007/01/codename-kolofonium-realease-date.html]
- ~~OpenWrt installation guide (Italian) and misc [http://blog.extreme-networking.com/]~~ Dead link — *tmomas 2016/03/16 10:14*
- ~~The Fonera in the Freifunk project, German [http://wiki.freifunk-hannover.de/Fonera_mit_OLSR]~~ English [http://wiki.freifunk.net/Fonera_with_OLSR_(English)]: comprehensive guide to flashing la fonera with Kamikaze.
- ~~Fonera Hacks - Numerous Tutorials and Guides [http://www.fonerahacks.com] Detailed tutorials with screenshots on how to flash the Fonera, unlock SSH, access Redboot, and other goodies.~~ Dead link — *tmomas 2016/03/16 10:18*
- making a FON router speak serial to Arduino [http://echodittolabs.org/blog/2007/12/making-fon-router-speak-serial-arduino]

Tags

How to add tags

FastEthernet, 1NIC, no switch, serial, integrated, 802.11bg, DetachableAntenna, 16RAM, 8Flash, MIPS, 4KEc, 5v powered, 12v powered