Contents

Gateworks Avila/Cambria

Thanks to

Brainslayer (Sebastian) to all the help send to me (and the patience he has) to explain many things about this topic (and others).

All **developer team** for this great firmware.

Gateworks for this device.

Notes about diferent firmware versions

Prerequisites

RedHat 5 or similar

I'm using CentOS 5 to write this.

I have installed **minicom**, **tftp-server** and **telnet** packages (and their dependences) and copied **zImage** and **root.fs** to **/tftpboot/**.

Then I configure /etc/xinetd.d/tftp file as this:

```
$ cat /etc/xinetd.d/tftp
# default: off
# description: The tftp server serves files using the trivial file transfer \
#       protocol.  The tftp protocol is often used to boot diskless \
#       workstations, download configuration files to network-aware printers, \
#       and to start the installation process for some operating systems.
service tftp
{
        socket_type             = dgram
        protocol                = udp
        wait                    = yes
        user                    = root
        server                  = /usr/sbin/in.tftpd
        server_args             = -s /tftpboot
        disable                 = no
```

```
        per_source              = 11
        cps                     = 100 2
        flags                   = IPv4
}
```

And verified that xinetd daemon is runing after the configuration:

```
$ service xinetd restart
Parando xinetd:                                            [  OK  ]
Iniciando xinetd:                                          [  OK  ]
$ service xinetd status
Se está ejecutando xinetd (pid 10353)...
```

Windows

See: Intall into GW2348

Accesing via telnet

By default you can access the device using a serial port cable (not null-modem) or using telnet.

Using TELNET

1) Configure your ethernet to have the 192.168.3.1/24 ip address

```
$ ip addr add 192.168.3.1/24 brd 192.168.3.255 dev eth0 label eth0:3
```

Where **eth0** is your ethernet interface and **eth0:3** is the alias (used if you have more than one IP assigned to the interface).

If you have an address assigned to the 192.168.3.0/24 subnetwork, delete it previously:

```
$ ip addr del 192.168.3.1/24 brd 192.168.3.255 dev eth0 label eth0:3
```

Will be usefull you clear the ARP entry for 192.168.3.2:

```
$ arp -d 192.168.3.2
```

And you add the correct MAC for that IP (you gain some milliseconds with this):

```
$ arp -s 192.168.3.2 00:D0:12:02:81:48
```

You must do it when you need to change cable to another GW234X device (for example).

2) Open a terminal window (a window better than a tab in some terminals programs) and send a ping to the GW234X.

```
$ ping 192.168.3.2
PING 192.168.3.2 (192.168.3.2) 56(84) bytes of data.
```

3) Open a new terminal window and prepare to do the telnet to the GW234X. Very important is that you **don't hit RETURN yet**.

```
$ telnet 192.168.3.2 9000
```

Where **192.168.3.2** is the default GW234X ip for access RedBoot and **9000** is the default port where RedBoot has a telnet daemon listening.

4) Go to the TELNET terminal and prepare de RETURN key and the CTRL+C combination.

5) Take in view the 2 terminal windows to see when the GW234X is answering the pings and plug-off/plug-on the power to the GW234X.

6) When the ping has the first answer pres RETURN and **very quickly** the CTRL+C keys.

7) If you were lucked, you can see the RedBoot Prompt:

```
$ telnet 192.168.3.2 9000
Trying 192.168.3.2...
Connected to 192.168.3.2 (192.168.3.2).
Escape character is '^]'.
== Executing boot script in 2.490 seconds - enter ^C to abort
^C
RedBoot>
```

8) If you were too slow hitting CTRL+C or hitting RETURN you will not have the RedBoot prompt and will need to go to step 3.

Using MINICOM

1) Configure MINICOM with the default values: 115200,8,N,1 without hardware/software flow control.

2) Save configuration (usefull to not reconfigure the access next time).

3) Connect a serial cable (DB9 MALE to DB9 FEMALE) from your PC to the GW234X.

4) Plug-off and plug-on the power to the GW234X.

5) Remember that you will need to hit CTRL+C keys, but now you, really, not need to do it quickly.

6) Play with RedBoot.

RedBoot

Commands summary

General

Here is it:

```
RedBoot> help
Manage aliases kept in FLASH memory
   alias name [value]
Set/Query the system console baud rate
   baudrate [-b <rate>]
Manage machine caches
   cache [ON | OFF]
Display/switch console channel
   channel [-1|<channel number>]
Compute a 32bit checksum [POSIX algorithm] for a range of memory
   cksum -b <location> -l <length>
Display disks/partitions.
   disks
Display (hex dump) a range of memory
   dump -b <location> [-l <length>] [-s] [-1|2|4]
Execute an image - with MMU off
   exec [-w timeout] [-b <load addr> [-l <length>]]
        [-r <ramdisk addr> [-s <ramdisk length>]]
        [-c "kernel command line"] [<entry_point>]
Manage FLASH images
   fis {cmds}
Manage configuration kept in FLASH memory
   fconfig [-i] [-l] [-n] [-f] [-d] | [-d] nickname [value]
Execute code at a location
   go [-w <timeout>] [-c] [-n] [entry]
Help about help?
   help [<topic>]
Display command history
   history
Set/change IP addresses
   ip_address [-l <local_ip_address>[/<mask_len>]] [-h <server_address>]
Load a file
   load [-r] [-v] [-d] [-h <host>] [-p <TCP port>][-m <varies>] [-c <channel_number>]
        [-b <base_address>] <file_name>
Dump information on PCI devices
   lspci
Compare two blocks of memory
   mcmp -s <location> -d <location> -l <length> [-1|-2|-4]
Copy memory from one address to another
   mcopy -s <location> -d <location> -l <length> [-1|-2|-4]
Fill a block of memory with a pattern
   mfill -b <location> -l <length> -p <pattern> [-1|-2|-4]
Network connectivity test
   ping [-v] [-n <count>] [-l <length>] [-t <timeout>] [-r <rate>]
        [-i <IP_addr>] -h <IP_addr>
Reset the system
   reset
Set/Read MAC address for NPE ethernet ports
   set_npe_mac [-p <portnum>] [xx:xx:xx:xx:xx:xx]
Swap bytes in 16-bit or 32-bit words in a block of memory
   swab -b <location> -l <length> [-2|-4]
Display RedBoot version information
   version
Display (hex dump) a range of memory
   x -b <location> [-l <length>] [-s] [-1|2|4]
```

fis

```
RedBoot> fis
*** invalid 'fis' command: too few arguments
Usage:
  fis create -b <mem_base> -l <image_length> [-s <data_length>]
      [-f <flash_addr>] [-e <entry_point>] [-r <ram_addr>] [-n] <name>
  fis delete name
  fis erase -f <flash_addr> -l <length>
  fis free
  fis init [-f]
  fis list [-d]
  fis load [-b <memory_load_address>] [-c] name
  fis lock [-f <flash_addr> -l <length>] [name]
  fis unlock [-f <flash_addr> -l <length>] [name]
  fis write -f <flash_addr> -b <mem_base> -l <image_length>
```

Basic procedures

Install / Recover dd-wrt firmware (>=v.24)

See the FLASHING.TXT. in the dd-wrt download page for the Avila firmware, for an uptodate information about the process.

See: Intall into GW2348 for an outdated explanation, but usefull for windows platforms.

Now, from some v24 rc's versions, only one image is needed and yes, use the explanation in this page to login into RedBoot and you only need these steps: ...on initial flash...

```
fis init -f
```

...or in case of recovery (else you will delete also the activation and all settings)...

```
fis init
```

...and

```
load -r -v -b 0x00800000 linux.bin
fis create linux
```

The process is this:

```
RedBoot> fis init -f
About to initialize [format] FLASH image system - continue (y/n)? y
*** Initialize FLASH Image System
... Erase from 0x50080000-0x50fe0000:    ..................................................
... Unlock from 0x50fe0000-0x51000000: .
... Erase from 0x50fe0000-0x51000000: .
... Program from 0x03fd0000-0x03ff0000 at 0x50fe0000: .
... Lock from 0x50fe0000-0x51000000: .

RedBoot> load -r -v -b 0x00800000 linux.bin
Using default protocol (TFTP)
|
```

```
Raw file loaded 0x00800000-0x00e75fff, assumed entry at 0x00800000

RedBoot> fis create linux
... Erase from 0x50080000-0x50700000: ..................................................
... Program from 0x00800000-0x00e76000 at 0x50080000: ..........................................
... Unlock from 0x50fe0000-0x51000000: .
... Erase from 0x50fe0000-0x51000000: .
... Program from 0x03fd0000-0x03ff0000 at 0x50fe0000: .
... Lock from 0x50fe0000-0x51000000: .
```

If you get locked flash problems like

```
RedBoot> fis create linux
... Erase from 0x50080000-0x50da0000: Err = a2
Can't erase region at 0x50080000: Device block is locked
```

try

```
fis unlock -f 0x50000000 -l 0x2000000
```

The last step is create your boot script using fconfig.

The boot script will be:

```
fis load linux
exec
```

Attention! If you have speed problems tranfsering data from one mpci slot to another, it might be a kernel problem. You should try this as boot script lines:

```
fis load linux
exec -c "console=ttyS0,115200 panic=10 mem=64M root=/dev/mtdblock2 rootfstype=squashfs,jffs2 no
```

You create your boot script as this:

```
RedBoot> fconfig
Run script at boot: true
Boot script:
.. fis load ramdisk
.. fis load zimage
.. exec
Enter script, terminate with empty line
>> fis load linux
>> exec
>>
Boot script timeout (100ms resolution): 25
Use BOOTP for network configuration: false
Gateway IP address:
Local IP address: 192.168.3.2
Local IP address mask: 255.255.255.0
Default server IP address: 192.168.3.1
Console baud rate: 115200
GDB connection port: 9000
Force console for special debug messages: false
Network debug at boot time: false
Default network device: npe_eth0
Update RedBoot non-volatile configuration - continue (y/n)? y
... Unlock from 0x50fe0000-0x51000000: .
```

```
... Erase from 0x50fe0000-0x51000000: .
... Program from 0x03fd0000-0x03ff0000 at 0x50fe0000: .
... Lock from 0x50fe0000-0x51000000: .
RedBoot>
```

You can now plug-off and plug-on your GW234x and enjoy dd-wrt on it.

Upgrade dd-wrt firmware (<v.24)

Use these steps for previous v24 prereleases that uses 2 images. The v24 only need one and is explained how to use it in the previous section.

Brainslayer explain to me the upgrade method very basically and it works very fine:

```
load -r -b 0x00800000 zImage
fis create linux
load -r -b 0x00800000 root.fs
fis create ramdisk
```

This is a snapshot of the process:

```
$ telnet 192.168.3.2 9000
Trying 192.168.3.2...
Connected to 192.168.3.2 (192.168.3.2).
Escape character is '^]'.
== Executing boot script in 2.220 seconds - enter ^C to abort
^C
RedBoot> load -r -b 0x00800000 zImage
Using default protocol (TFTP)
Raw file loaded 0x00800000-0x00976107, assumed entry at 0x00800000
RedBoot> fis create linux
An image named 'linux' exists - continue (y/n)? y
... Erase from 0x50080000-0x50280000: .................
... Program from 0x00800000-0x00976108 at 0x50080000: ............
... Unlock from 0x507e0000-0x50800000: .
... Erase from 0x507e0000-0x50800000: .
... Program from 0x01fe0000-0x02000000 at 0x507e0000: .
... Lock from 0x507e0000-0x50800000: .
RedBoot> load -r -b 0x00800000 root.fs
Using default protocol (TFTP)
Raw file loaded 0x00800000-0x00cfafff, assumed entry at 0x00800000
RedBoot> fis create ramdisk
An image named 'ramdisk' exists - continue (y/n)? y
... Erase from 0x50280000-0x507a0000: ....................................
... Program from 0x00800000-0x00cfb000 at 0x50280000: ....................................
... Unlock from 0x507e0000-0x50800000: .
... Erase from 0x507e0000-0x50800000: .
... Program from 0x01fe0000-0x02000000 at 0x507e0000: .
... Lock from 0x507e0000-0x50800000: .
RedBoot>
```

WiFi Configurations

Wireless modes

AP (Access Point)

The wifi radio interface is configured to operate in MASTER mode and synchronizes all wireless clients (managed radios) that connect to it. You need to configure the acces parameters, including security settings, in order to allow clients to connect (or not connect) to it. You must configure the ESSID, authentication and cipher parameters.

The wifi interface is connected to an ethernet interface, either through bridging or routing, to allow access to the wireless LAN.

At the moment, the WDS configured using this mode don't use the wireless encryption options, it will use a simple WDS mode without any encryption type. See the B.S. explanation here: BS explanation about encryption under WDS modes. To have this, the devices must be configured explicitly in WDS-AP and WDS-STA modes.

Client bridge

The same as AP but the wifi interface is turned into a MANAGED mode and you use the device to connect many computers using LAN to the WIFI LAN.

You must configure the same values in ESSID, auth and cipher to allow connect it to the APs in the wifi LAN.

The WIFI MAC is proxied to allow connect many PCs using only one device to the wifi LAN.

If you use the special ESSID **any** you will allow to connect to any wireless network that shares the auth and cipher configuration with your device.

Client

Also known as **Routed Client**.

The wifi interface is turned into a MANAGED mode and will connect to any AP that shares the ESSID, authentication and cipher configuration with it.

The wifi interface has the role of another ethernet interface and will be used as the WAN port of the router.

AdHoc

Really is a **routed adhoc client**. The same as client routed but the wifi interface only will connect to another ad-hoc devices that shares the same auth and cipher configuration.

AdHoc bridge

Is not impletented by DD-WRT firmware (and don't know why).

The ideal implementation is to be used as client bridge without ARP PROXY, to allow the point to point links to eat all the band.

Here is some information about this:

http://madwifi.orghttp://wiki.dd-wrt.com/wiki/UserDocs/AhdemoInterface

WDS Station and WDS AP

This is a mode that allow the devices to be configured in WDS configurations and allow encrypted links between the WDS-STA devices and the WDS-AP device.

In the WDS-STA devices are very usefull the **ScanList** parameter that allow the device to search channels only in the configured ranges.

See: http://madwifi.orghttp://wiki.dd-wrt.com/wiki/UserDocs/WDSBridge

Please: Complete this if you know more links or more references about this mode and more accurate info about this.

Drivers Parameters

Important notes

As **BrainSlayer** has remarked to me, the madwifi driver inside dd-wrt is not the madwifi-ng.

I remember he had a trunk from an old madwifi driver and began to work into it to make it stable, full featured and useful from the firmware point of view.

The explanation about the parameters are based on the oficial madwifi documentation and many of them our own experiences configuring and testing GW234X using the diferent parameters.

You are warned about these, if somebody knows better about how each parameter works: please feel free to modify this page and insert the correct information.

Antenna Gain

Setting the proper antenna gain in this field will ensure that you are in compliance with local regulatory requirements for radio frequency emissions. It will allow the firmware to determine the maximum allowed TX power for your country. The combination of TX power and antenna gain determine whether or not you comply with local EIRP (Effective Isotropic Radiated Power).

Turbo Mode

Turbo Mode uses 40MHz signal bandwidth rather than 20MHz signal bandwidth. In this mode Atheros devices achieve a doubling (108Mbps) of the over-the-air data rate. Two modes are available, Static and Dynamic Turbo. Certain channels in the 2.4GHz and 5GHz bands are reserved for operating in these modes.

See: http://madwifi.orghttp://wiki.dd-wrt.com/wiki/UserDocs/TurboMode and http://madwifi.orghttp://wiki.dd-wrt.com/wiki/UserDocs/802.11a_channels

Notes:

1) When you configure GW23XX using this mode, you can't see it using kismet or any other sniffing tool. I haven't played with this very much, but I think you can put the **kis0** interface in your sniffer machine in this mode and adapt the kismet.conf to allow kismet to see these types of AP's.

2) This is not a standard mode, be warned.

3) Before you use it in outdoor links, use kismet with standard channels to have the antennas oriented. First use a standard channel near the "non standard" channel to use kismet (or the tool you use to adjust antenna position): after that, you can enable the turbo mode.

Extended range

Atheros' eXtended Range technology (XR) consists of a signal processing architecture that stretches the coverage of a WLAN by embedding separate optimized designs for both high performance/high signal-to-noise ratio environments and long range/low signal-to- noise ratio environments. XR is able to maintain standards compliance while providing extended range performance.

```
- Up to -105dBm receive sensitivity
- Processes extremely weak signals
- Supports additional transmission rates equivalent to 3, 2, 1, 0.5, 0.25 Mbps
- Extends support for signals beyond 802.11 boundaries
- Standards-interoperable
```

Super G Compression

Link-level hardware compression more efficiently utilizes the wireless connection to further maximize bandwidth. This compression is implemented on a per frame basis and affects only data frames. The concept is similar to that used in common data file compression utilities such as WinZip. This also requires an AP that supports compression. When used with fast framing, the theoretical maximum throughput is ~40mbps using standard channel widths and spacing.

Super G Fast Framing

Super G further provides throughput benefits through fast frames. While bursting increases the number of frames transmitted in a given transmission opportunity, Fast framing allows for more information per frame to be transmitted. Rather than restrict frames to the standard size, the frame size can be negotiated between a transmitter and receiver, thus maximizing efficiency via less overhead. This requires an AP that supports fast frames. When used with compression, the theoretical maximum throughput is ~40mbps using standard

channel widths and spacing.

Outdoor Band

The **ddwrt atheros driver** allows you to select 'only indoor' channels or 'only outdoor' channels with this option. Regulatory requirements determine which frequencies are available in each operating mode. Frequencies that are available in one operating mode may or may not be available in the other operating mode. In order to comply with regulatory domain requirements you should select the appropriate operating mode and stick with the frequencies that are available.

Diversity

Antenna diversity can be realized in several ways. Depending on the environment and the expected interference, designers can employ one or more of these methods to improve signal quality. In fact, multiple methods are frequently used to further increase reliability.

- Spatial Diversity ? Spatial diversity employs multiple antennas, usually with the same characteristics, that are physically separated from one another. Depending upon the expected incidence of the incoming signal, sometimes a space on the order of a wavelength is sufficient. Other times much larger distances are needed. Cellularization or sectorization, for example, is a spatial diversity scheme that can have antennas or base stations miles apart. This is especially beneficial for the mobile communications industry since it allows multiple users to share a limited communication spectrum and avoid co-channel interference.

- Pattern Diversity ? Pattern diversity consists of two or more co-located antennas with different radiation patterns. This type of diversity makes use of directive antennas that are usually physically separated by some (often short) distance. Collectively they are capable of discriminating a large portion of angle space and can provide a higher gain versus a single omni-directional radiator.

- Polarization Diversity ? Polarization diversity combines pairs of antennas with orthogonal polarizations (i.e. horizontal/vertical, ± slant 45°, Left-hand/Right-hand Circularly Polarized etc). Reflected signals can undergo polarization changes depending on the media. By pairing two complimentary polarizations, this scheme can immunize a system against polarization mismatches that would otherwise cause signal fade. Additionally, such diversity has proven valuable at radio and mobile communication base stations since it is less susceptible to the near random orientations of transmitting antennas.

- Transmit/Receive Diversity ? Transmit/Receive diversity uses two separate, co-located antennas for transmit and receive functions. Such a configuration eliminates the need for a duplexer and can protect sensitive receiver components from the high power used in transmit. This method can be used in DD-WRT to great effect when complying with regulatory requirements for long distance connections. For instance, selecting the 'main' or 'primary' port of a radio card for transmission allows for an antenna/power configuration which complies with local EIRP restrictions, while selecting the 'aux' or 'secondary' port of a radio card for reception allows for a much higher gain receive antenna to receive weak signals.

Channel Width

Channel Width or 'bandwidth' is an important concept to understand when determining how a base station or 'AP' is to be deployed. The majority of 802.11 deployments use a standard 20MHz channel width in order to connect to standard devices such as laptops and PDAs. If the user is not connecting to these standard devices,

the full potential of radio communications becomes available. Simply stated, if you reduce the transmitted bandwidth, you increase the amount of effective power, increase receive sensitivity, and reduce available bit-rate deliverable to the other end. What does this mean? If you reduce your TX bandwidth to 5MHz or 10MHz you can extend your reach much further, but at the cost of reduced throughput. In certain situations where a 'long reach' is required, this may be your only option. Minimizing bandwidth can also help overcome 'noisy' environments where frequency interference is an issue. It also has the added advantage of increasing the number of available TX channels.

OFDM SIFS Time

Don't know yet. Any physical parameter to adjust the modulated signal.

OFDM Preamble Time

Don't know yet. Any physical parameter to adjust the modulated signal.

Sensitivity Range (ACK Timing)

Adjust this when you have point to point links (WDS) to allow some driver internals optimization.

ScanList

In automatic scanning modes, such as 'Client' or 'Station' modes this is the field that represents the list of frequencies to use for scanning.

Enter a list of frequencies (in MHz) separated with spaces. You can use ranges as well with values separated by a hyphen.

For example, put your channel in **Auto** mode and in **ScanList** put **2437-2447 5180-5200**. This will allow your device to scan from channels 4 to 6 and from channels 36 to 40. This is useful with WDS-STA mode and Client modes.

B.S. notice that this is a very useful parameter when using the SuperChannel extension with Atheros devices.

In SuperChannel mode with a 'Mixed' mode of operation, it will most likely be necessary to create a scanlist. Depending on the type of radio card and manufacturer, many, many frequencies will be available for scanning. Limiting the scanlist to frequency ranges that you are interested in will greatly improve connection time.

For more information about channels and frecuencies take a look at:

- The Wikipedia
- MadWifi Documentation

SuperChannel extension

Do not confuse this with Atheros SuperA/G modes.

SuperChannel increases the number of channels that you can use allowing more Bandwidth to be used (not more speed, be careful not to make this mistake!).

In many places there are many wireless devices operating and it is a bit difficult to configure a robust wireless link without frequency interference. If you have purchased a SuperChannel extension from DD-WRT, then enabling this feature will allow you to access additional frequencies supported by the radio card.

You must take care to operate within your local regulatory requirements. If necessary, you must request a license to operate on licensed frequencies. **Be warned** that the use of many of these aditional channels will break your regulatory domain requirements in your country and if you were discovered perhaps (surely) you will be penalized.

SuperChannel allows you to use extended channels. In 5 MHz steps you can select any channel from 2.3 - 2.7 GHz and 4.9 - 6.1 GHz. Please be aware that even though the SuperChannel extension shows that these frequencies are available, there are hardware limitations (frequency filters) that may or may not be installed on the radio card which eliminate the possibility of using certain frequencies.

SuperChannel is compatible with Turbo, allowing you to use Turbo mode on any of these extended channels.

SuperChannel bypasses the regulatory domain in all countries, but you still can operate within a regulatory domain by disabling this feature.

Thanks to **BrainSlayer** for his explanation and remarks about this extension.

SuperA/G mode

The first aproximation in break the 54Mps restriction in A/G modes with Atheros chipsets where to use a centered channel (channel 6 in G) and a aditional channel to double the speed (doubling the bandwidth without overlaping the channels).

In this mode you only can select the centered channel and the another channel will be selected dynamically. You only can use some channels in this mode, selecting, really, the centered channel.

Supposition: I think that centered channels depend on the regulatory domain selected (it has its logic), but I don't know certainly.

I remember that SuperA/G modes are used principally in AP modes, don't know yet if it can be used in WDS links or not. If anybody know, please, put here.

Turbo mode born to allow the user to sum the bandwidth (and the speed) of the channels used in a bigger channel (has a doubled bandwidth) and select that channel statically, allowing the user to control better the bandwidth.

NOTE: Don't know if ddwrt support this mode, I think this is an obsoleted mode.

Configuration examples

Normal AP WDS configuration between 2 GW2347 and 1 GW2348

This case we'll connect 3 LAN using 2 GW2347 and 1 GW2348 and they will be transparent for the direfent Ethernet subnets.

```
GW2347_0 - - - - - GW2348 - - - - - - GW2347_1
   |                  |                   |
  LAN                LAN                 LAN
```

Because this is a bit extense document (with some screenshots) is writen into this page: Normal AP WDS configuration between 2 GW2347 and 1 GW2348

WDS-AP/WDS-STA WDS configuration between 2 GW2348-2 and 1 GW2348-4

It's the same case, but using WDS-AP and WDS-STA specific modes.

```
GW2348_0 - - - - - GW2348_1 - - - - - - GW2348_2
   |                  |                   |
  LAN                LAN                 LAN
```

Because this is a bit extense document (with some screenshots) is writen into this page: WDS-AP/WDS-STA WDS configuration between 2 GW2348-2 and 1 GW2348-4

Standing Network configuration

I put this here to remember things that I want to write (or see writen by somebody).

GW234X interfaces

Here I put the interfaces defined by default in GW234X device when they are booted with a dd-wrt v24 firmware flashed.

GW2347

When you use 1 atheros miniPCI card, the router will configure these physical interfaces:

ixp0: The first ethernet interface.

wifi0: The first Atheros card. This is the real interface and is used for management purposes.

ath0: The first wifi defined interface over wifi0 real interface.

ath0.N: The (N+1)th wifi defined interface over wifi0 real interface.

ath0.wdsN: The (N+1)th WDS link defined virtual interface over ath0 interface.

(obsoleted) **wdsath0.N**: The (N+1)th WDS link defined virtual interface over ath0 interface.

lo: loopback interface

imq0: Intermediate Queueing Device (see http://www.linuximq.net/). For QoS control purposes (General In).

imq1: Intermediate Queueing Device (see http://www.linuximq.net/). For QoS control purposes (Generao Out).

bond0: The first bonding defined interface.

teql0: Don't know yet.

GW2348-2/4, GW2358-4

When you use 2 atheros miniPCI card, the router will configure these physical interfaces:

ixpN: The (N+1)th ethernet interface.

wifiN: The (N+1) Atheros card. This is the real interface and is used for management purposes.

athN: The first wifi defined interface over wifiN real interface.

athN.M: The (M+1)th wifi defined interface over wifiN real interface.

athN.wdsM: The (M+1)th WDS link defined virtual interface over athN interface. *(obsoleted)* **wdsathN.M**: The (M+1)th WDS link defined virtual interface over athN interface.

lo: loopback interface

imq0: Intermediate Queueing Device (see http://www.linuximq.net/). For QoS control purposes (General In).

imq1: Intermediate Queueing Device (see http://www.linuximq.net/). For QoS control purposes (Generao Out).

bond0: The first bonding defined interface.

teql0: Don't know yet.

Networking concepts

Routing

Bridges

Bonding

See: http://www.linux-foundation.org/en/Net:Bonding

VLAN

Configuration examples