Details of the Router Boards

We have two types of Router Board:

1. Mikrotik RB433AH
   i. 128 MB of RAM and 680s MHz CPU
   ii. Plugged with 2 R52Hn Radio, this supports 802.11 a/b/g/n.
   iii. One Serial Port and Three Ethernet Port and a power supply adapter.
2. Mikrotik RB411
   i. 32 MB of RAM 300MHz CPU
   ii. Plugged with one R52Hn Radio, this supports 802.11 a/b/g/n.

Components required to power up a Board

1. PoE - Power Over Ethernet Adapter
2. Power Supply Adapter
3. Two straight LAN cable.
4. RS-232 Serial cable to access it through ZoC terminal in PC.

Normally, Router boards comes with pre-install router OS provided by Mikrotik. You can access it through a well-known tool Win box also provided by Mikrotik.

You need to connect the PoE ODU port to board and RJ45 port to your system LAN.

Installing openWRT on Router Board

1. For compiling and getting the code follow the document.
2. After compiling prepare your system for DHCP boot.
3. First of all you have to boot the board over Ethernet for that you need to access the board through ZoC terminal and select boot device "o →e" and change the boot protocol to DHCP i.e. "2".
4. If the board doesn't get the DHCP server than give the ip configuration again to the source system.
5. After successful net boot prepare the source system for tftp server. This is listed in the document too.
6. Now to get permanent OS in memory of the board you need to type command "wget2nand http://192.168.0.1" in the board. Two problem may occur-
   i. Network Unreachable: For this you need to again configure the ip and restart the tftp srever.
   ii. File not found: The tar.gz file that the wget2nand is downloading may present with a different name in the source directory. So change the name to the file it is requesting.

Note: Delete the /tmp/wget2nand file every time you retype the command. "rm –rf /tmp/wget2nand"

7. After installing the OS in NAND memory reboot the board and select boot device NAND only i.e. "o " option.
8. To get required packages you connect it to internet through Ethernet.
9. To configure proxy you need to put the following line in /etc/opkg.conf
   i. option http_proxy http://proxy4.nehu.ac.in:3128/
   ii. option ftp_proxy ftp://proxy4.nehu.ac.in:3128/
10. Edit the file /ete/resolve.config to put nameserver as 172.16.1.1.
11. Now do a ping test to the proxy4 server. If all fine you can update typing "opkg update".
12. The packages you will need yafc,ssh,ipef. You can install it by typing "opkg install packagename"

Creating the sendraw and sendwan command

The source file and Makefile is available in the laptops for both command.

1. Create directory in /trunk same name as source file.
2. Copy the respective c files and Makefile in each directory.
3. Compile using make command in the same directory individually.
4. Make directory with name of the command in /trunk/fedds/packages/utils.
5. Copy the Makfile that's given with the source in folder utils to each respective folder. Don't forget to change the user name where necessary in Makfiles.
6. Type the following command
   i. ./scripts/feeds update –i
   ii. ./scripts/feeds install sendraw
   iii. ./scripts/feeds install sendwlan
   iv. make menuconfig
   v. Find your package in Utils-→ sendraw and Utils→ sendwlan
   vi. make package/sendraw/compile
   vii. make package/sendwlan/compile

After successful compilation you will get two .ipk file in /bin/ar7xx/packages/ starting with sendraw and sendwlan. To install this two ipk you have to download the ipk using yafc ftp client in board.

After downloading install using "opkg install *.ipk".

The Code Structure and module installation

Source code of the openWRT OS will be inside the trunk directory only. The code is layered in different directory as MAC and N/L. The MAC part will be inside **/trunk/build_dir/target-mips_34kc_uClibc-0.9.33.2/linux-ar71xx_mikrotik/compat-wireless-2014-01-23.1/net/mac80211/**

All the plays will be in this field only. Remember it!.

Changes can be made at any source file but the main compilation will be executed in **/trunk** directory only using **make V=s** command. Upon compilation you will get the following updated module

Name of Module	Directory
ath9k.ko	/compat-wireless-2014-01-23.1/net/wireless/ath/ath9k/
ath9k_common.ko	/compat-wireless-2014-01-23.1/net/wireless/ath/ath9k/
ath9k_hw.ko	/compat-wireless-2014-01-23.1/net/wireless/ath/ath9k/
ath.ko	/compat-wireless-2014-01-23.1/net/wireless/ath/
compat.ko	/compat-wireless-2014-01-23.1/compat
cfg80211.ko	/compat-wireless-2014-01-23.1/net/wireless/
mac80211.ko	/compat-wireless-2014-01-23.1/net/mac80211/

To install the modules in Board do the following steps

1. Access the board using ssh "ssh root@ip address of the board.
2. Get the modules in board using the yafc ftp client.

3. First you need to remove the already loaded modules using rmmod in sequence ath9k→ath9k_common→ath9k_hw→ath →mac80211→ cfg80211→compat.
4. Install the modules using insmod in a reverse sequence.
5. After installing you need to put your wireless interfaces in monitor mode to see the magic of your changes.

For your convenience the shell scripts has been created you just need to run it simply.

Code details

The three main C files in the directory are

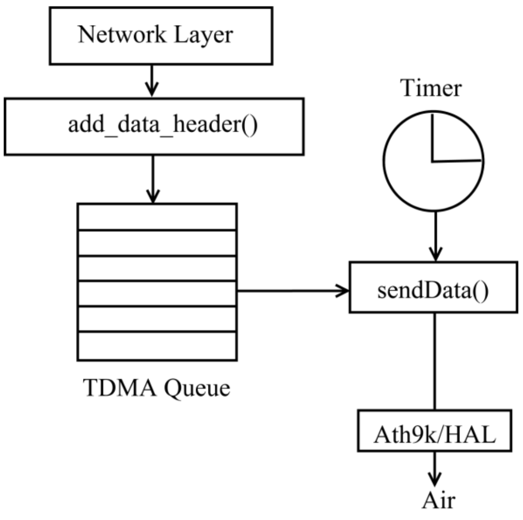
tx.c – Mainly responsible for transmission of packet and MAC protocol encapsulation

rx.c – For reception of packet and verification.

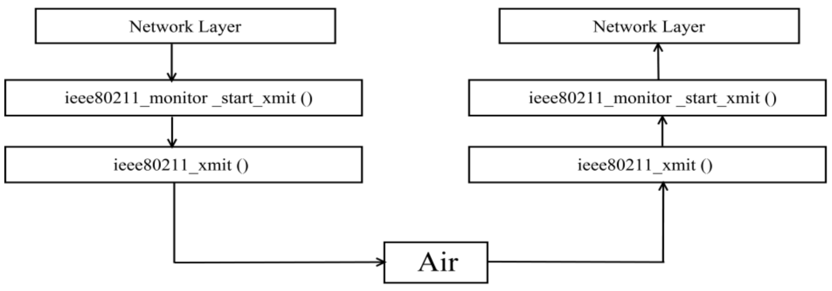
main.c – Module initialization functions resides here.

Since our work on transmission and reception side we have separately draw code in a header file with respect to every source file and then include it with the source. The files are **2C-tx.h, 2C-rx.h** and **2C-main.h**.

TDMA Work



As you can see the above figure, it's same as how it happens in the code. Now let's see how and where we have to hit.


Network Layer
ieee80211_monitor _start_xmit ()
ieee80211_xmit ()
Network Layer
ieee80211_monitor _start_xmit ()
ieee80211_xmit ()
Air

So our target is to abstract the CSMA/CA and implement the TDMA over the existing code. For that we will proceed our implementation in iee80211_monitor_start_xmit() in tx side. The detail of the implementation is provided in other documents.

The three main structures from the network kernel are sk_buff, sdata and channel

sk_buff is the socket buffer that contains the original packet information. ***data** pointer points to the packet.

sdata contains the information of interface.

chan contains the channel related information.

These three structures we will always need to send any packet in air whether it is control or data packet.

<u>Changes made in tx.c</u>

1. We have made changes and commented some line in **iee80211_monitor_start_xmit()** to allow flow of packet in monitor mode. You can navigate in this function and will see the changes.
2. For sending control packet from MAC layer we need the three structures to send in air. Since this structure are created in upper layer so we have created three global structure namely globalskb, globalsdata and globalchan and copy an instance of every structure from upper layer into it.

<u>Queuing in MAC layer</u>

For queuing we maintain a software queue using linked list data structure. The packet from the network layer is handed to **ieee80211_monitor_start_xmit()** passing skb and network device information . We queue the both information in **tdma_queue**. The a function called **sendData()** will extract from the queue and passes it to **ieee80211_xmit()** which transmit the packet in air. We maintain a timer called **test_timer()** which will call the **sendData()** function periodically.

Note: The details of the variable, macros and logic is commented in every line of the code where required. So please don't panic.

For more information please kindly look at the two documents along with this.

Other supporting documents are

1. Driver.pdf→ Will help you to understand the implementation and a brief of code details.
2. Flashing OpenWrt onto RouterBoard 433AH.pdf
3. Step-By-Step-Instruction-To-Run-Apps-On-FlexRoad-HW.en.pdf → Making application for openWRT

Other Important file and scripts

1. documents-exports-2014-08-25.zip → The three C and three header file code with mlti-hop support.
2. sendraw.tar.gz and sendwlan.tar.gz→ Raw packet sending application source.
3. copy.sh → Will copy all the modules from different directory to one directory. To be run on source system.
4. install.sh → Will install all the modules in board . To be run on router board.
5. monitor.sh→ Will make the selected board to monitor mode. To be run on router board.

************Good Luck ************