Flashing OpenWrt onto RouterBoard 433AH

OpenWrt is a Linux distribution primarily targeted at routing on embedded devices. OpenWrt can be configured using the Linux command-line interface.

The following references were used in this lab:

http://wiki.openwrt.org/doc/howto/build

https://forum.openwrt.org/viewtopic.php?id=15201

https://help.ubuntu.com/community/CiscoConsole

http://open80211s.org/trac/wiki/OpenWrtBuilding

*If server already has trunk file skip to Setup Serial Console making sure that the DHCP server has been configured to give the routerboard's MAC address a static IP address.

Netboot

This is done on the host computer, not the RouterBOARD. The host computer must be connected to the internet so the files can be downloaded. The following packages should be installed now while connected to the internet. They will be configured and used throughout the process:

```
sudo apt-get install dnsmasq
sudo apt-get install minicom
sudo apt-get install mini-httpd
```

Getting the OpenWrt Trunk:

1. Set up the trunk using the following code:

```
git clone git://nbd.name/openwrt.git ~/trunk/
cd ~/trunk/
./scripts/feeds update
```

2. Use the following code to check what packages are missing. Then use apt-get to install the missing packages. Figure 1 provides the name of the package depending on what distribution of Linux is being used.

```
make menuconfig
apt-get install <missing packages in table>
```

Prerequisite	Debian	Suse	Red Hat	OS X	Fedora	NetBSD
asciidoc	asciidoc	asciidoc	asciidoc	?	asciidoc	?
bash	?	?	?	?	?	bash
binutils	binutils	binutils	binutils	?	binutils	?
bzip2	bzip2	bzip2	bzip2	?	bzip2	?
fastjar	fastjar	fastjar	libgcj	?	libgcj	?
flex	flex	?	?	?	flex	?
g++	g++	gcc-c++	gcc-c++	?	gcc-c++	?
gcc	gcc	gcc	gcc	?	gcc	?
getopt	?	?	?	?	?	getopt
GNU awk	gawk	gawk	gawk	?	gawk	?
gtk2.0-dev	libgtk2.0-dev	?	gtk2-devel	?	gtk2-devel	?
intltool-update	intltool	intltool	intltool	?	intltool	?
jikes	—	jikes	?	?	—	?
libz, libz-dev	zlib1g-dev	zlib-devel	zlib-devel	?	zlib-devel	?
make	make	make	?	?	make	gmake
ncurses	libncurses5-dev	ncurses-devel	ncurses-devel	?	ncurses-devel	?
openssl/ssl.h	libssl-dev	libopenssl-devel	openssl-devel	?	openssl-devel	?
patch	patch	patch	?	?	patch	?
perl-ExtUtils-MakeMaker	perl-modules	perl-ExtUtils-MakeMaker	perl-ExtUtils-MakeMaker	?	perl-ExtUtils-MakeMaker	?
python2.6-dev	python2.6-dev	?	?	?	?	?
rsync	rsync	rsync	?	?	rsync	?
ruby	ruby	ruby	?	?	ruby	?
sdcc	sdcc	sdcc	?	?	sdcc	?
unzip	unzip	unzip	?	?	unzip	?
wget	wget	wget	wget	?	wget	?
working-sdcc	—	?	?	?	—	?
xgettext	gettext	?	?	?	gettext	?
xsltproc	xsltproc	libxslt	?	?	libxslt	?

Unfortunately not all dependencies are checked by `make config`.

Figure 1. List of packages required to make the boot image.

3. Once all the appropriate packages have been added type make menuconfig. A GUI should pop up asking for the configuration of OpenWRT. This menu allows for packages to be added and removed from the image that is being built. Change the settings to the following:

```
Target System
        [*] Atheros AR7xxx/AR9xxx
Subtarget
        [*] Devices with NAND flash (mostly Mikrotik)
Target Profile
        [*] Default Profile (No WiFi)
Target Images
        [*] ramdisk
        [*] tar.gz
Base System
        [*] Base-files
        [*] Busybox
        [*] dnsmasq
        [*] hotplug2
        [*] mtd
        [*] opkg
        [*] wireless-tools
Kernel Modules
        Network Support
                [*] kmod-8021q
        Wireless Drivers
                [*] kmod-ath9k
Network
        SSH
                [*] openssh-client
                [*] openssh-server
        Web Servers/Proxies
                [*] lighttpd
                        [*] lighttpd-mod-cgi
                        [*] lighttpd-mod-auth
[*] hostapd
```

4. Run the following command to make the boot image (.elf).

```
make world
```

Note: This will probably take at least an hour.

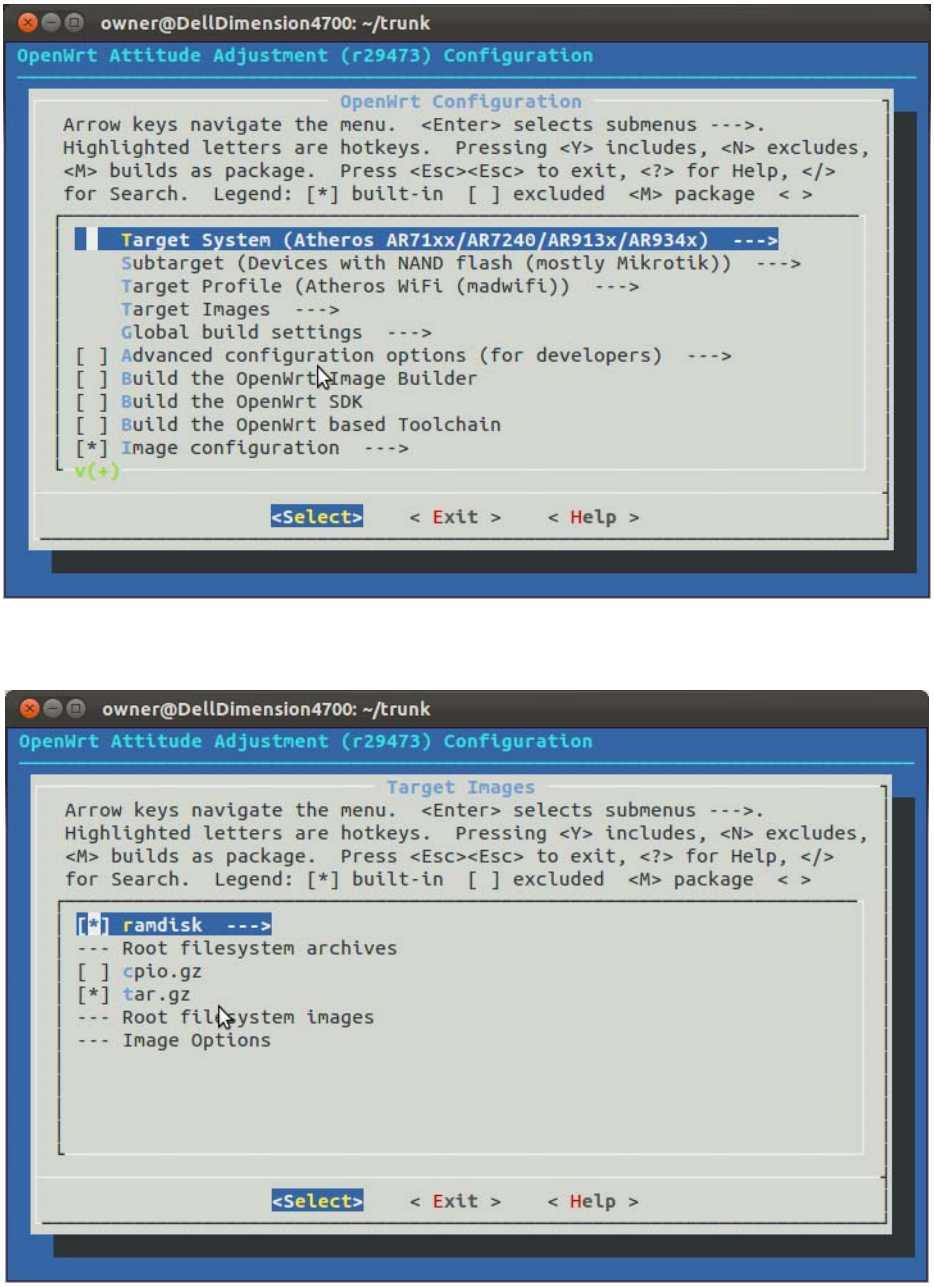

Figure 2. Screenshots from the menuconfig of OpenWRT.

5. After the image is done making, make a directory that the image will be booted from. Then make a shortcut from where the image actually is to the tftp folder using the following commands:

```
sudo mkdir –p /tftpboot

sudo ln –sf /home/estadium/trunk/bin/ar71xx/openwrt-ar71xx-
nand-vmlinux-initramfs.elf /tftpboot/openwrt.elf
```

6. All files have been downloaded. In order for the RouterBOARD to boot over Ethernet, the Ethernet port on the host computer must be configured. Run the following commands to set up the Ethernet port:

```
sudo /etc/init.d/network-manager stop
sudo /etc/init.d/network restart
sudo ifconfig eth0 192.168.0.1 netmask 255.255.255.0
```

Setting up the DHCP and TFTP Server

7. Make changes to the /etc/dnsmasq.conf as follows:

   - only listen on the interface which is directly connected to the RouterBoards WAN (PoE) port. e.g.

     ```
     interface=eth0
     ```

   - allow a DHCP range, e.g.

     ```
     dhcp-range=192.168.0.50,192.168.0.150,12
     ```

   - allocate an IP address to the board (replace the XX with the MAC address of your board!)

     ```
     dhcp-host=00:0C:42:XX:XX:XX,192.168.0.60
     ```

   - enable dnsmasq's built-in TFTP server

     ```
     enable-tftp
     ```

   - set the root directory for files available via FTP.

     ```
     tftp-root=/tftpboot
     ```

   - set filename (produced above) and tftpd server for BOOTP, e. g.

     ```
     dhcp-boot=/tftpboot/openwrt.elf,boothost,192.168.0.1
     ```

8. Finally restart dnsmasq

   ```
   sudo /etc/init.d/dnsmasq restart
   ```

When the RouterBOARD is powered on, the host computer will give the assigned IP address to the RouterBOARD. This will allow the RouterBOARD and the host computer to communicate with each other and transfer files.

Setting up a Serial Console

9. Use the following command to find what the name the serial port is. After the word serial and some numbers, there will be ttyXX. This is the name of the serial port and will be needed later.

```
dmesg | grep tty
```

10. Run minicom with the command below. Choose Serial Port Setup and change the name of the serial port to the name that was found in step 11. Figure 3 shows the minicom setup.

```
sudo minicom –s
```

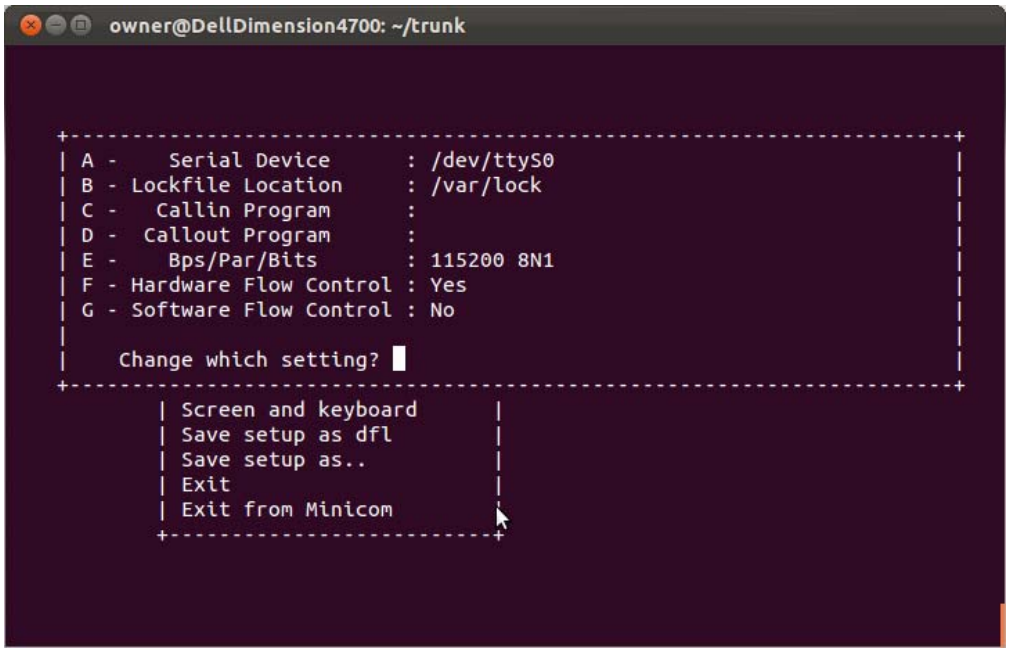

Figure 3. Screenshot of the minicom setup.

11. Choose Exit. The serial port is all set up. Connect the serial cable from the RouterBOARD to the computer. Also, connect the RouterBOARD from the Ethernet port with the MAC address added to the DHCP server to the Ethernet port on the computer that the DHCP server is listening on.
12. Connect the RouterBOARD to power. Immediately after connecting power, text should start showing up in the minicom window. Press enter before two seconds to interrupt the boot sequence on the RouterBOARD.
13. Press o to change the boot device and then select e which tells the RouterBOARD to boot over Ethernet. Press x to leave the setup and let the RouterBOARD reboot. If all has been done right, the output should look similar to figure 4.

```
RouterBoard 433AH



CPU frequency: 680 MHz

 Memory size: 128 MB



Press any key within 2 seconds to enter setup..

trying bootp protocol..... OK

Got IP address: 192.168.6.101

resolved mac address 4E:80:00:00:00:00

Gateway: 192.168.6.1

transfer started ..... transfer ok, time=7.16s

setting up elf image... OK

jumping to kernel code

Linux version 2.6.26.7 (joerga@thinkpad) (gcc version 4.1.2) #2 Mon Nov 10 11:23:37 CET 2008

console [early0] enabled

...
```

Figure 4. Output from successfully netbooted routerboard.

14. Leave the board netbooted for the permanent installation.

Permanent Installation

Setting up an HTTP server

1. Make the appropriate changes to /etc/default/mini-httpd:
    - Start daemon

```
START=1
```

2. Make the appropriate change to /etc/mini-httpd.conf:

   - On which host mini_httpd should bind

   ```
   host=192.168.0.1
   ```

   - Run in chroot mode

   ```
   chroot
   ```

   - Where are the web files stored

   ```
   data_dir=/home/owner/trunk/bin/ar71xx/
   ```

3. Restart the mini-httpd web server

   ```
   sudo /etc/init.d/mini-httpd restart
   ```

Building the Permanent Image

4. Change into the directory trunk that was used to make the first image. All the settings are the same except the target images. Run menuconfig and use the following settings:

```
cd ~/trunk/

make menucofig

          Target System
        [*] Atheros AR7xxx/AR9xxx
    Subtarget
        [*] Devices with NAND flash (mostly Mikrotik)
    Target Profile
        [*] Default Profile (No WiFi)
    Target Images
        [*] tar.gz
        [*] squashfs
    Base System
        [*] Base-files
        [*] Busybox
        [*] dnsmasq
        [*] hotplug2
        [*] mtd
        [*] opkg
        [*] wireless-tools
    Kernel Modules
        Network Support
            [*] kmod-8021q
        Wireless Drivers
            [*] kmod-ath9k
    Network
        SSH
            [*] openssh-client
            [*] openssh-server
        Web Servers/Proxies
            [*] lighttpd
                [*] lighttpd-mod-cgi
                [*] lighttpd-mod-auth
    [*] hostapd
```

5. Make the permanent image:

```
make world
```

Install OpenWrt into Flash

6. On the routerboard which is still netbooted, press enter to get a shell prompt.
7. Run the following line of code which will delete the flash memory and copy the OpenWRT image into it

```
wget2nand http://192. 168.0.1
```

8. When the transfer has completed, type reboot. The routerboard will reboot. Press enter to interrupt the boot sequence. Choose option o to change the boot device. Then choose option o to boot from the NAND only. Choose option x and let the routerboard reboot.
9. After the routerboard has reboot, press enter to get a shell prompt. The file rc.local is the script that automatically runs at boot. In order to configure the wireless cards, the hardware encryption needs to be turned off. This will be added to the rc.local file so it is done everytime the routerboard boots using the following code:

```
vi /etc/rc.local
rmmod ath9k
insmod /lib/modules/2.6.39.4/ath9k.ko nohwcrypt=1
ifconfig eth0 192.168.<networkID>.10<Node#>
ifconfig eth0 netmask 255.255.255.0
```

10. Save and exit the file (Shift + zz) and reboot the routerboard. The routerboard now has OpenWRT running. Using the iwconfig command, the wireless cards should be visible. They are down when the routerboard boots so they will not show up in ifconfig. Use the ifconfig command to bring them up.

Configuring RouterBOARD

1. Install nano. Nano is a text editor for the terminal that is much more user friendly than vi.

```
opkg update
opkg install nano
```

2. Change the hostname and the timezone

```
nano /etc/config/system
hostname Node<NetworkID>-<Node#>
timezone
```

3. Set password (e$tad465) for root, which will be used for SSH. Configure OpenSSH to accept connections from root.

```
passwd

nano /etc/ssh/sshd_config

PermitRootLogin yes
```

4. Reboot the RouterBOARD. If all went well, you should be able to SSH into the RouterBOARD from the host computer by running the follow command from the terminal on the host:

```
ssh root@192.168.<networkID>.10<Node#>

Or

ssh root@Node<networkID>-<Node#>
```