BRINGING ROUTERS AND MODEMS TOGETHER IN STYLE

ROOter Build Systems

DIY ROOTER

Create an ROOter Build System

To create a OpenWrt build system that can be used to create ROOter images you need a computer that is running Linux.

This can be a Virtual Machine running in Windows or Linux running on a computer. The recommended Linux distros to use for this are Linux Mint, Ubuntu 18.04 or Debian. ROOter images are made using Linux Mint 20. This computer must have Internet access in order to download the files needed to create an OpenWrt build system.

Using the File Manager of your Linux distro, create a folder that will contain the build system. It is possible to have multiple build systems resident in this folder if you want to build images with different OpenWrt versions.

We will name this folder ***OpenWrt*** *but it can be called anything.*

Using the File Manager go to the ***OpenWrt*** *folder and open a Terminal session. How this is done depends on the distro used. In Linux Mint you right click and chose Open in Terminal.*

The first thing that needs to be done is to add all the extra packages to your distro that are required when building an image. This is done from the Terminal by entering the following commands.

For Linux Mint

sudo apt-get install build-essential subversion git-core libncurses5-dev zlib1g-dev quilt sudo apt-get install gawk flex quilt libssl-dev xsltproc libxml-parser-perl jshon

For Ubuntu

sudo apt -y install build-essential libncurses5-dev python unzip gawk git curl jshon

For Debian 11

sudo apt install build-essential bash binutils bzip2 flex git-core g++ gcc util-linux gawk sudo apt install help2man intltool libelf-dev zlib1g-dev make libglvnd-dev pkg-config sudo apt install libncurses5-dev libssl-dev patch perl-modules git ncurses-dev libz-dev sudo apt install python2-dev wget gettext xsltproc zlib1g-dev zip unzip libssl-dev sudo apt install python2 python3-dev libelf-dev subversion gettext gawk wget curl sudo apt install rsync perl unrar rar jshon

Once you have the required packages installed you can create the build system. At the Terminal enter the following lines, waiting for each to complete before entering the next one.

This will take a bit of time and requires an Internet connection.

There are a number of different build systems for ROOter depending on which OpenWrt version you want to use. Different routers use different build systems that are matched to the specifications of the router. Newer build systems usually require more processor power and flash/RAM so you have to chose the correct build system for your router.

The first step is to clone the build system from a Git. The following is a list of the command you need to run to do this for each OpenWrt version.

Version 21.02	**git clone https://github.com/ofmodemsandmen/ROOterSource2102 source2102**
Version 22.03	**git clone https://github.com/ofmodemsandmen/ROOterSource2203 source2203**
Version 23.05	**git clone https://github.com/ofmodemsandmen/RooterSource2305 source2305**
Version Master	**git clone https://github.com/ofmodemsandmen/SourceMaster sourcemaster**
Version Main	**git clone https://gitlab.com/ofmodemsandmen/SourceMain sourcemain**

Check the firmware downloads at ROOter Autobuilds to see which build system is used for your router. The suffix at the end of the file name will tell which build system is used.

- AB19 - OpenWrt 19.07
- AB21 - OpenWrt 21.02
- AB22 - OpenWrt 22.03
- AB23 - OpenWrt 23.05
- ABSM - OpenWrt Master
- ABSmn - OpenWrt Main

Both Master and Main are Snapshot versions of OpenWrt and are newer that the official 23.05 version. Main is about 6 months newer than Master. Both Master and Main require a router with more than 16meg of flash memory because of the increased size of the kernel version being used.

Run one of the above commands to create a build system in the folder whose name is at the end of the command (ie. source2102, sourcemain, etc). This will take some time as it must download the build system and set it up. Once this has completed you can close the Terminal session as the build system is ready to use.

Creating a Firmware

There are two steps to creating a new image, defining the router and the packages in the image and compiling the image. To do this use the File Manager to go to the build system folder and open a Terminal session there.

If you wish to customize the firmware by adding extra packages you must edit the router's configuration file. If you are just making a standard ROOter firmware then you can skip this step. If you are customizing the firmware then to find the name of the configuration file run this in the Terminal.

./build

This will give you a list of the configuration files that this build system supports. For the Main build system it will like this.

Supported Models are :

LBR20	RASPPI5	RASPPI5USB	BPI4
AXT1800	AX1800	AW1000	X3000
X86-64	RASPPI4	RASPPI4USB	RASPPICM4
RASPPICM4USB	BPI3MINI	MT2500	TR3000
WG156	Z8102	BPI4POE	Z800
RASPPI	RASPPI2	RASPPI3	RASPPI3USB

*The names of the configuration file will, in most cases, indicate which router they are for. You can also look in the **/configfiles/template** folder and see the name of the files there. They all start with ".config_" and what follows is the name you want.*

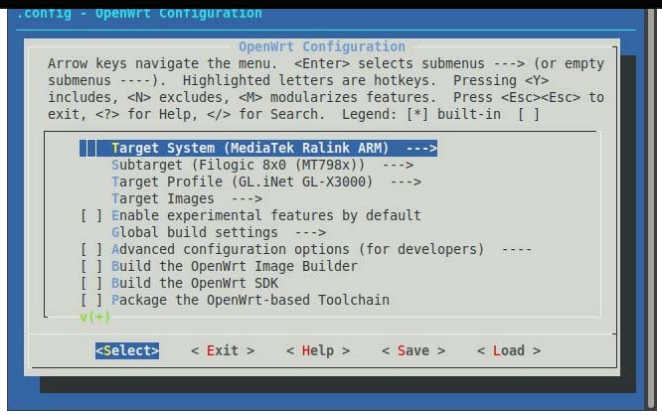
To customize the configuration file you enter this in the Terminal.

./menu config_name

*where **config_name** is one of the above names without the ".config_" prefix. The name should be in lower case like*

./menu x3000

This will open the standard OpenWrt configuration window in the Terminal so you can add new packages to the firmware.



When you exit, if you chose to save the changes, the configuration file in ***/configfiles/template*** *will be updated.*

The next step is to compile the firmware using the router's configuration file. This is done by running this in the Terminal.

./build config_name

where ***config_name*** *is one of the above names without the ".config_" prefix. The name should be in lower case like*

./build x3000

This will compile the firmware and rename the resulting files into the ROOter format, then place a Zipped archive of them into the ***/images*** *folder ready for your use.*

The first time you build a firmware with a new build system it needs to download all the packages needed for the process. This is a large number of files, 4GB for the Main build system, so it takes a while to finish running. The speed depends a lot on the computer's processor speed and your Internet speed. After this the compile will be faster. Once you have made a firmware you can also speed things up by using more of the processor's cores. ***This can not be done before you build the first firmware as the build will error.*** *To utilize more of the processor's core and build using multiple threads you must edit the build file in the root folder. At the start of this text file you will see this.*

```
#!/bin/sh

#core="-j7"
core=""
```

This forces the build to use only a single core and thread. By changing it to this.

```
#!/bin/sh

core="-j7"
#core=""
```

you will allow it to use 6 cores. Change the 7 to a value which is one more than the maximum cores you want to allow the build system to use. On a processor with 3 cores this value would be 4 to utilize all the cores. Doing this will greatly speed up the compile time.

If you find the build ends with an error then the first thing to do is to clear all the build files and start again. This is done by running this in the Terminal.

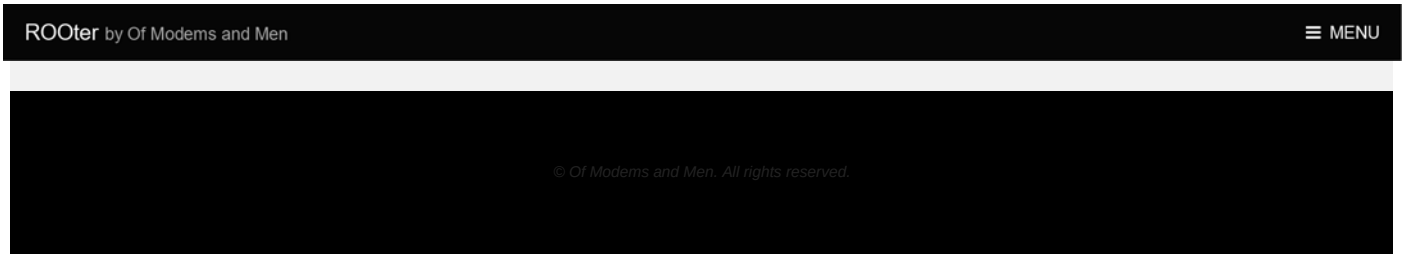
make dirclean

This removes all of the already compiled files and starts the build system all over again which can solve a lot of build errors that seem to appear for no good reason.

There is also a limit to how many extra packages that can be added to a firmware for a 16meg router. If too many packages are added the firmware will exceed the maximum size of the router's flash and no firmware will be created, causing the build to end in an error.

Also realize that as you move from the 19.07 build system to the Master/Main build systems the Linux kernel used by OpenWrt increases in size by about 1meg per version. This makes the firmware from the newer versions bigger and, eventually, they get too big for the available flash memory.

In newer versions the amount of free flash memory required for the firmware to run also increases. In 19.07 about 100meg of free flash was the minimum needed for the firmware to run correctly while in Master/Main it is about 250meg. This affects the number and size of any extra packages you may wish to add to a 16meg router.


ROOter by Of Modems and Men
MENU
© Of Modems and Men. All rights reserved.