

## **Building ROOter Images**

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 build-essential gawk flex quilt libssl-dev xsltproc libxml-parser-perl
```

For Ubuntu

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

For Debian

```
sudo apt install build-essential git unzip ncurses-dev libz-dev libssl-dev openssl-1.0-dev  
sudo apt install build-essential python python3-dev python3.5 libelf-dev subversion  
sudo apt install build-essential gettext gawk wget curl rsync perl
```

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.

First clone the build system from GitHub into a folder named *router19076*. This folder name can be changed to anything by changing the name in the command.

```
git clone https://github.com/ofmodemsandmen/router19076 router19076
```

This will take some time as it must download the build system and set it up. Then move into the *router19076* folder and update all the build packages.

```
cd router19076  
./scripts/feeds update -a  
./scripts/feeds install -a
```

Once this has completed you can close the Terminal session as the build system is ready to use.

## **Creating a New Image**

There are two steps to creating a new image, defining the router and the packages in the image and compiling the image.

Use the File Manager to go to the **OpenWrt** folder and open a Terminal session there

All information about the image is contained in a file named **.config** in this folder. Note that there is a period at the start of the name. For each image you wish to create you must generate this file.

Contained in this file are the router's SOC (processor type), router make and model and a list of all the packages you wish to include in the image.

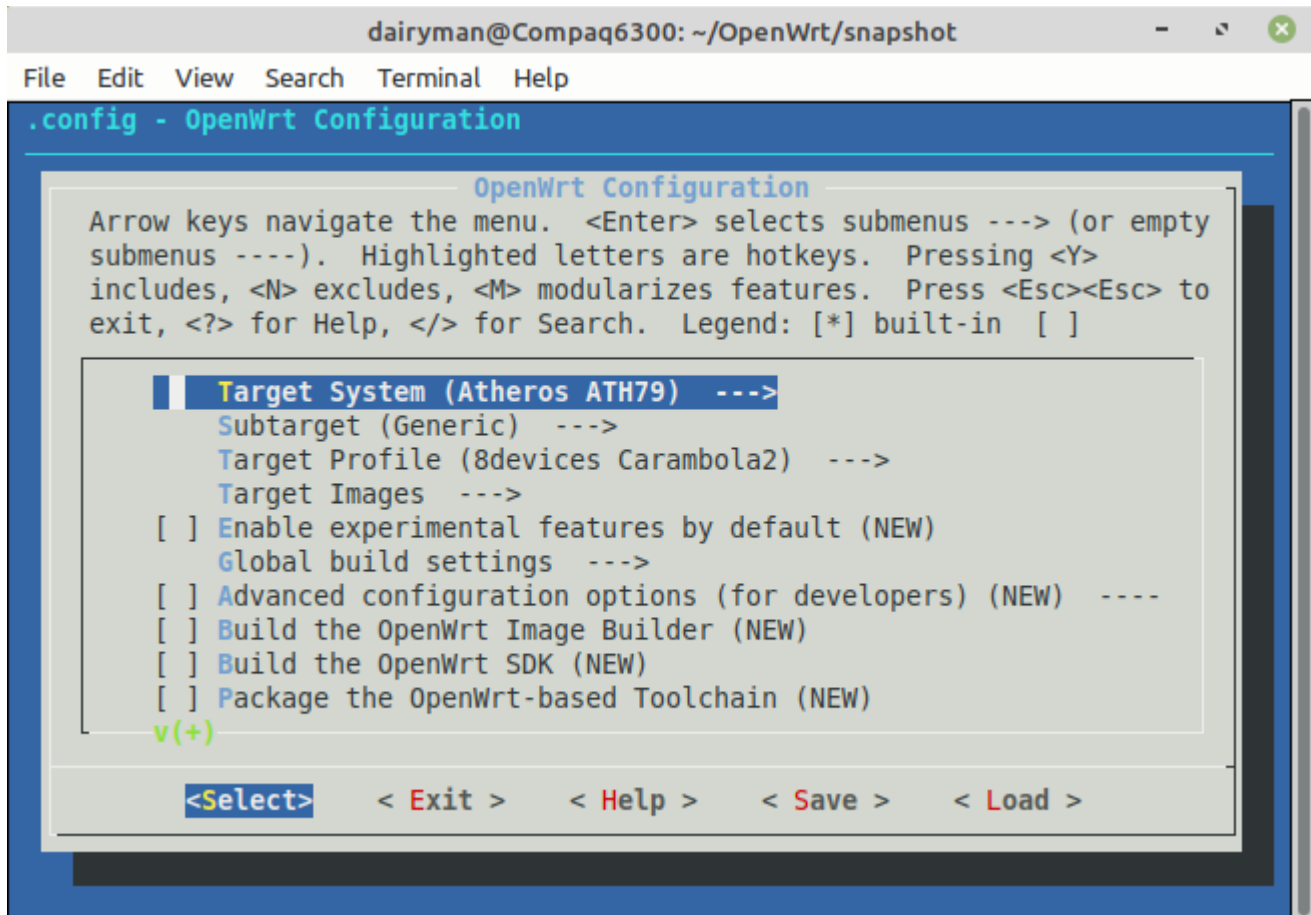
You create the **.config** file by entering the following in the Terminal.

```
make menuconfig
```

If the **.config** file already exists in the folder you can now edit it. If it doesn't exist then it will be created.

If you are creating a **.config** file for a router that is different from the one defined by the existing **.config** file then you should delete the old file before running the **make menuconfig** command. This will stop any errors creeping into the definition.

After running the command you should see this in the Terminal window.



The screenshot shows a terminal window titled "dairyman@Compaq6300: ~/OpenWrt/snapshot". Inside the terminal, the "OpenWrt Configuration" menu is displayed. The menu is titled ".config - OpenWrt Configuration" and "OpenWrt Configuration". It contains instructions: "Arrow keys navigate the menu. <Enter> selects submenus ---> (or empty submenus ----). Highlighted letters are hotkeys. Pressing <Y> includes, <N> excludes, <M> modularizes features. Press <Esc><Esc> to exit, <?> for Help, </> for Search. Legend: [\*] built-in [ ]". The menu items are: "Target System (Atheros ATH79) --->", "Subtarget (Generic) --->", "Target Profile (8devices Carambola2) --->", "Target Images --->", "[ ] Enable experimental features by default (NEW)", "Global build settings --->", "[ ] Advanced configuration options (for developers) (NEW) ----", "[ ] Build the OpenWrt Image Builder (NEW)", "[ ] Build the OpenWrt SDK (NEW)", "[ ] Package the OpenWrt-based Toolchain (NEW)". At the bottom, there are navigation options: "<Select>", "<Exit>", "<Help>", "<Save>", and "<Load>".

This window is navigated by using the keyboard. The **left** and **right** arrow keys move you across the commands at the bottom of the screen. The **up** and **down** arrow keys move you through the menu entries.

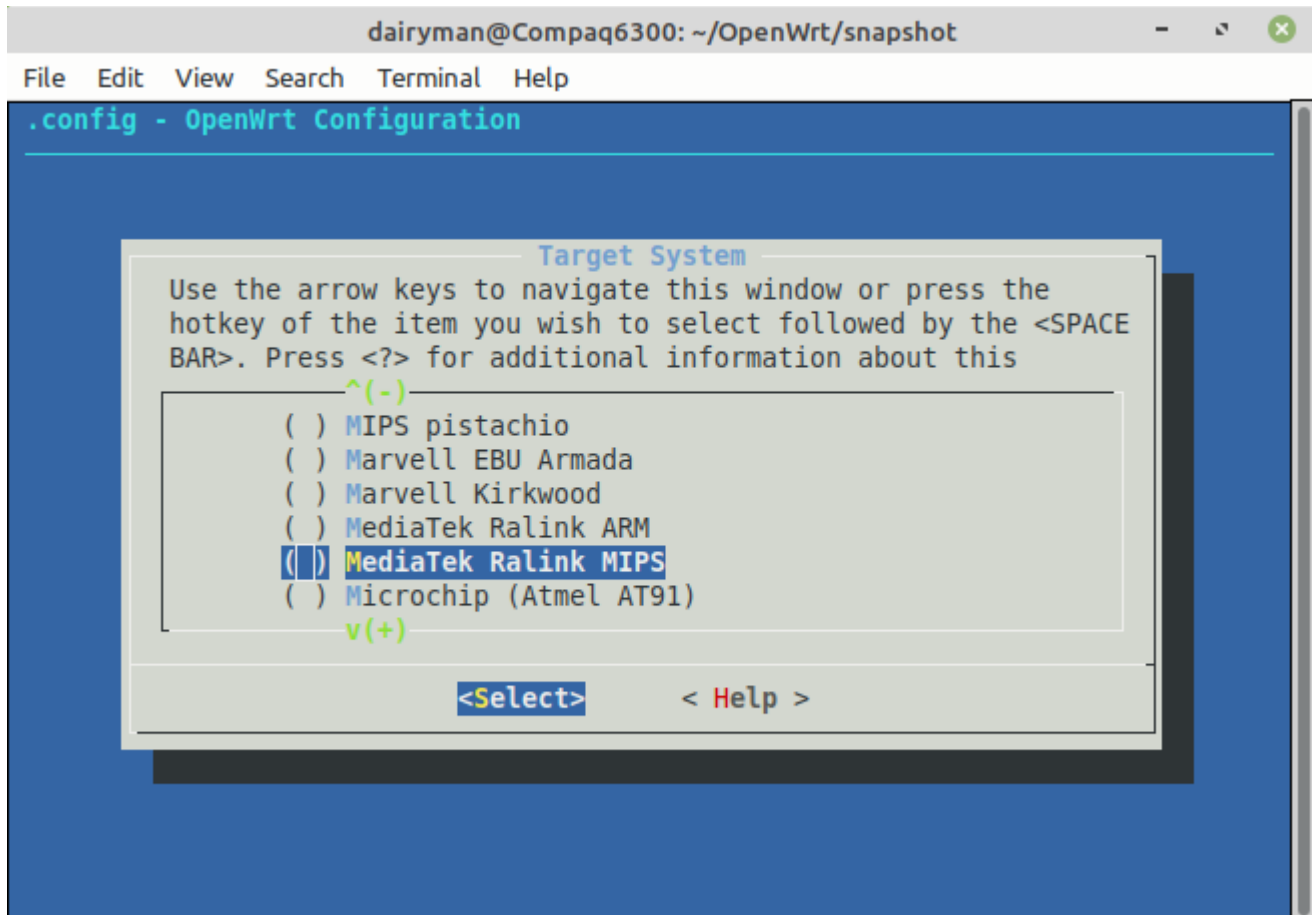
Pressing **Enter** will take you to the selected submenu if the highlighted bottom command is **<Select>**. If the highlighted bottom command is **<Exit>** then you will go to the previous submenu or, if at the main menu, will exit the Configuration program

If the **.config** file was changed or is new you will be asked if you wish to save it.

## Configuring the Target

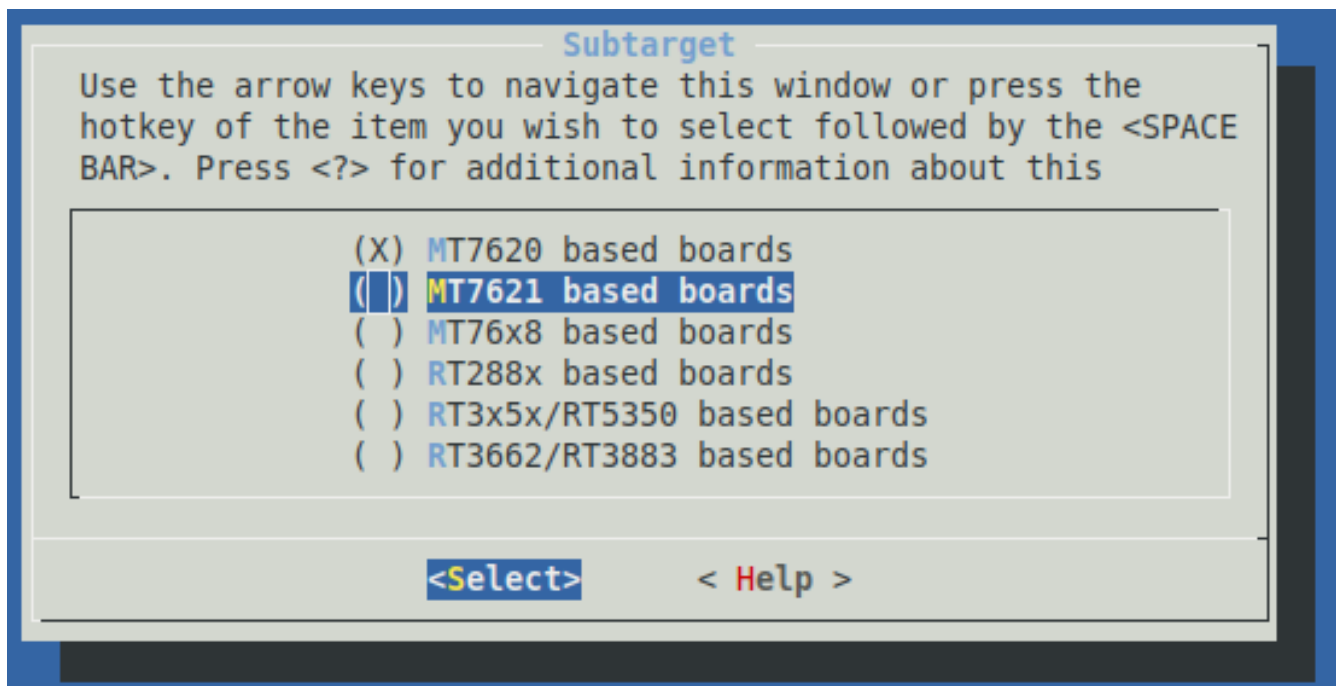
Since every image created by the build system is for a specific router you must tell the Configuration program what router you are building for. This is done by setting the correct **Target System**, **Subtarget** and **Target Profile**.

The **Target System** is used to select the SOC or processor type of the router. Use the arrow keys to highlight this menu entry and then press **Enter**.



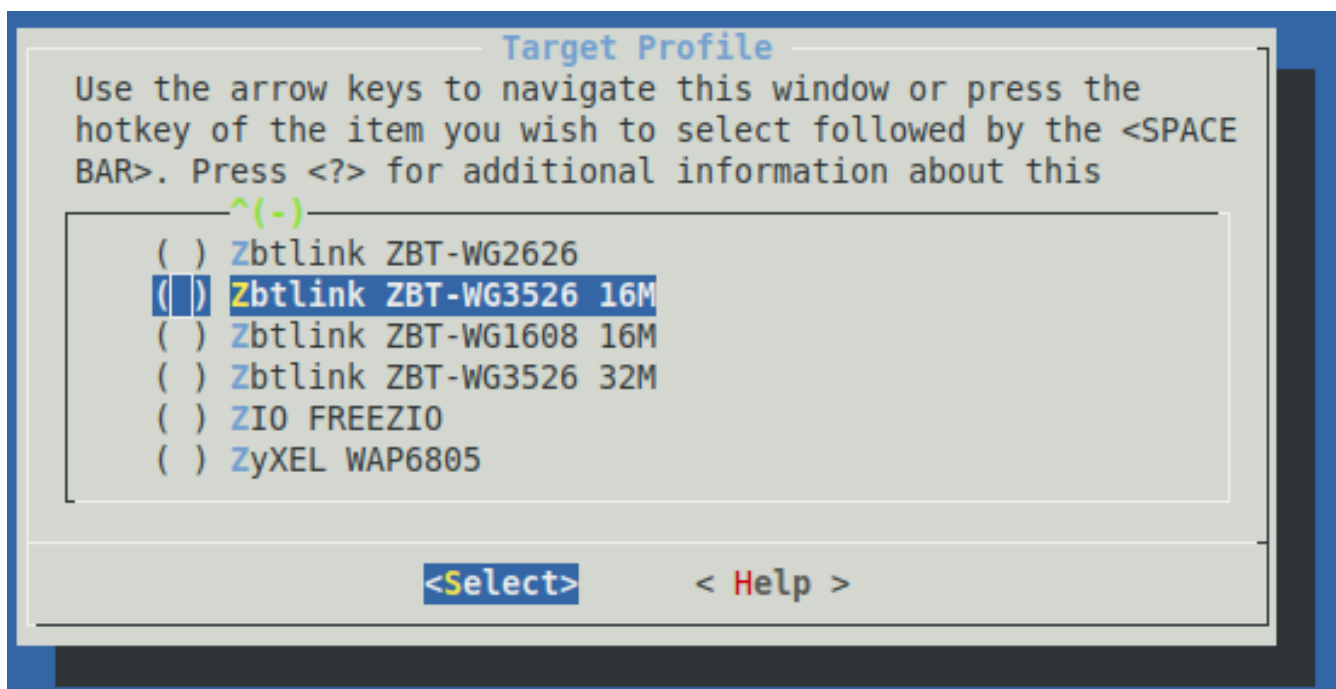
Use the arrow keys to scroll up and down this menu until you find the desired **Target System** then press **Enter** to select it.

The **Subtarget** menu (when it is present) is used to select among the different types of this processor.



Again, use the arrow keys to scroll up and down this menu until you find the desired **Subtarget** and press **Enter** to select it.

Finally, the **Target Profile** is used to select the router make and model.



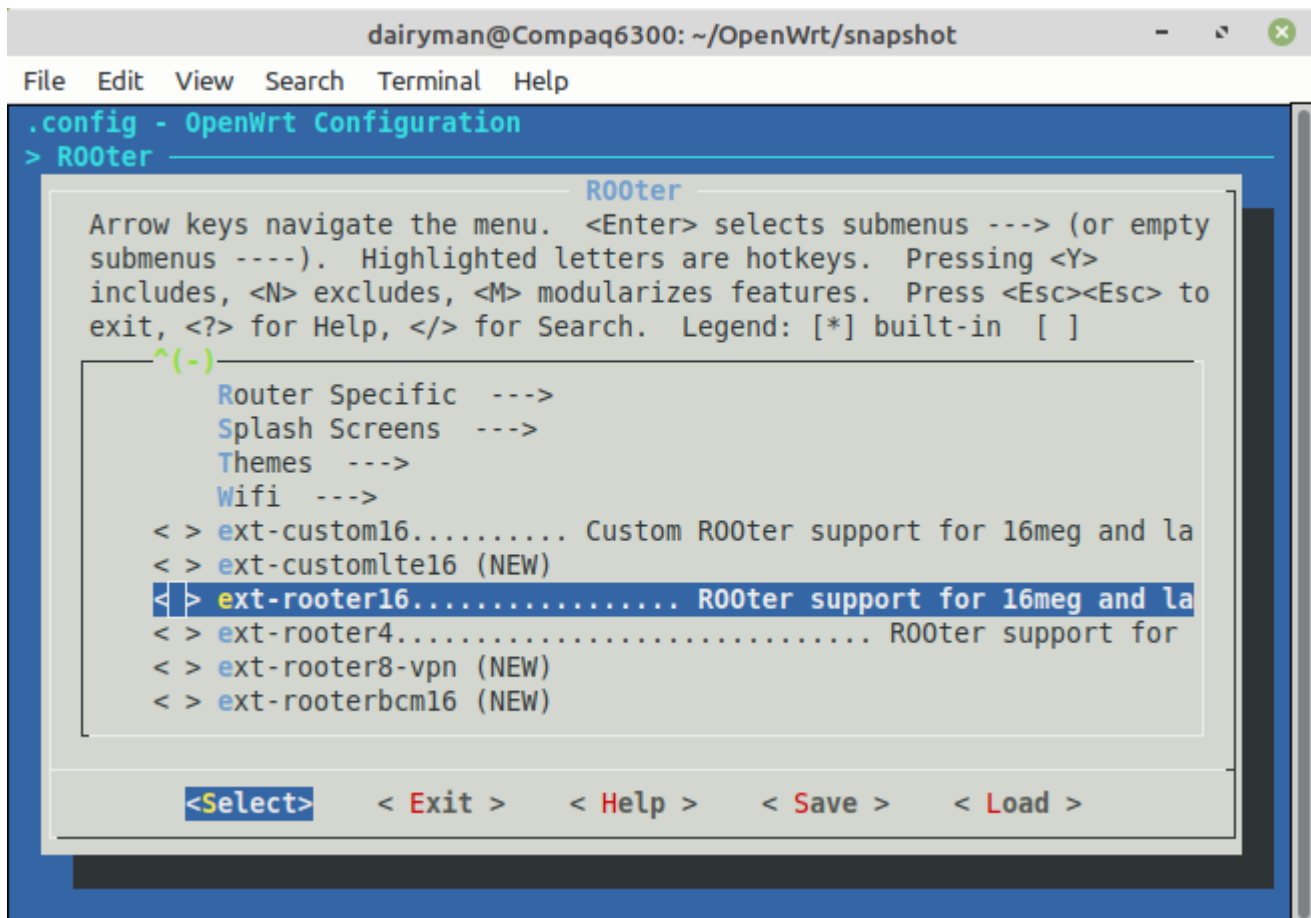
Use the arrow keys to scroll up and down this menu until you find the desired router make and model and then press **Enter** to select it.

When you have completed the selection of the router make and model you are ready to add the extra packages to the image that make up ROOter.

## Package Selection

With target router selected we can now select which packages we want to have included in the firmware. These packages will add extra features not found in a basic OpenWrt firmware.

To build a ROOter firmware, scroll down to the **ROOter** menu entry and press **Enter**.



```
dairyman@Compaq6300: ~/OpenWrt/snapshot
File Edit View Search Terminal Help
.config - OpenWrt Configuration
> ROOter

                                ROOter
Arrow keys navigate the menu. <Enter> selects submenus ---> (or empty
submenus ----). Highlighted letters are hotkeys. Pressing <Y>
includes, <N> excludes, <M> modularizes features. Press <Esc><Esc> to
exit, <?> for Help, </> for Search. Legend: [*] built-in [ ]

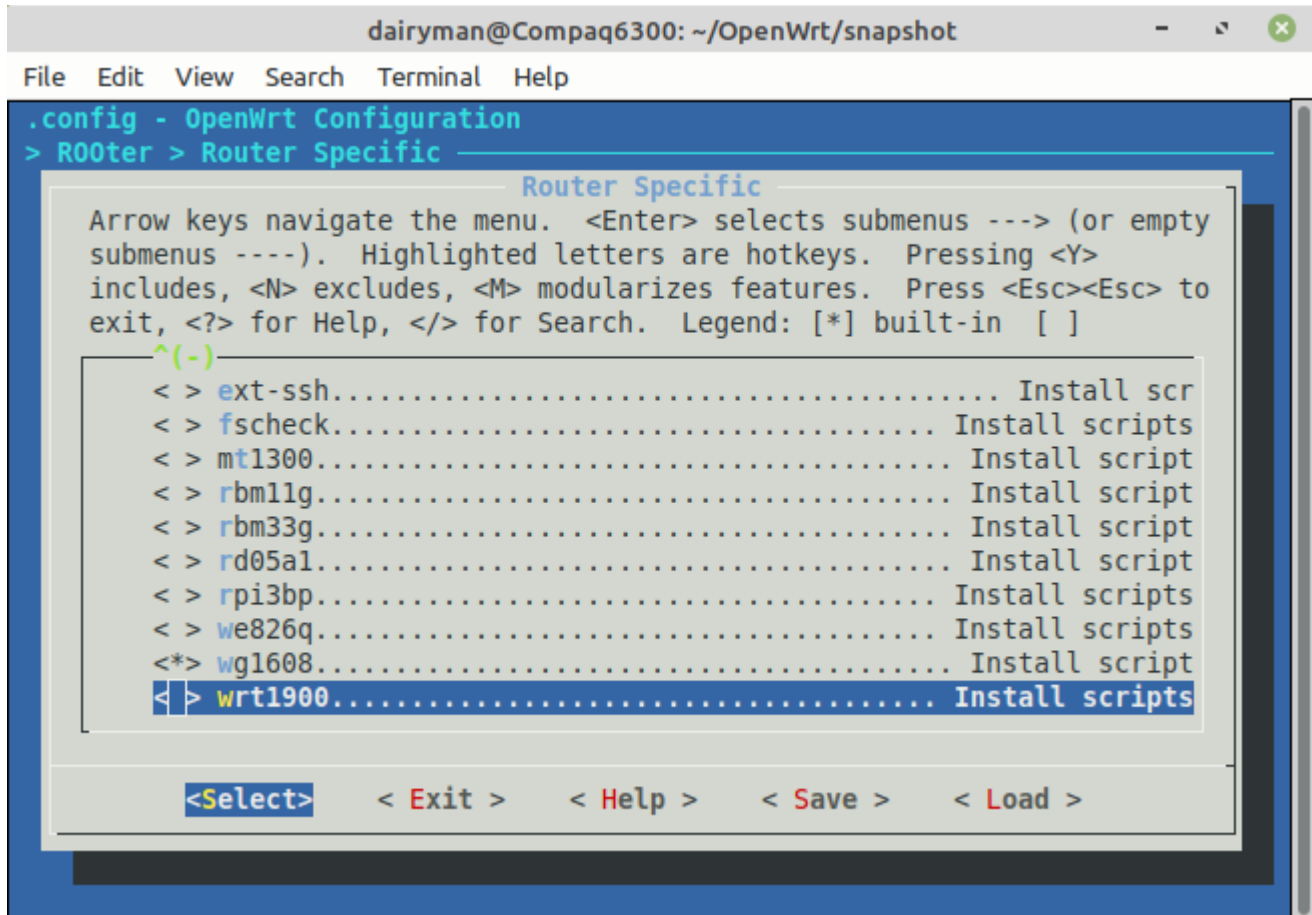
^(-)
Router Specific --->
Splash Screens --->
Themes --->
Wifi --->
< > ext-custom16..... Custom ROOter support for 16meg and la
< > ext-customlte16 (NEW)
[*] < > ext-rooter16..... ROOter support for 16meg and la
< > ext-rooter4..... ROOter support for
< > ext-rooter8-vpn (NEW)
< > ext-rooterbcm16 (NEW)

<Select> < Exit > < Help > < Save > < Load >
```

Select the ROOter package that is designed for your router, based on the amount of flash memory and processor type. You do this by pressing **Y**. To deselect a package press **N**. A **\*** will appear to the left of a package that has been selected.

In a later section we will describe what each ROOter package contains.

For some routers you must also select extra ROOter packages designed specifically for those routers. To do this, scroll to the **Router Specific** menu and press **Enter**.



```
dairyman@Compaq6300: ~/OpenWrt/snapshot
File Edit View Search Terminal Help
.config - OpenWrt Configuration
> ROOter > Router Specific

Router Specific
Arrow keys navigate the menu. <Enter> selects submenus ---> (or empty
submenus ----). Highlighted letters are hotkeys. Pressing <Y>
includes, <N> excludes, <M> modularizes features. Press <Esc><Esc> to
exit, <?> for Help, </> for Search. Legend: [*] built-in [ ]

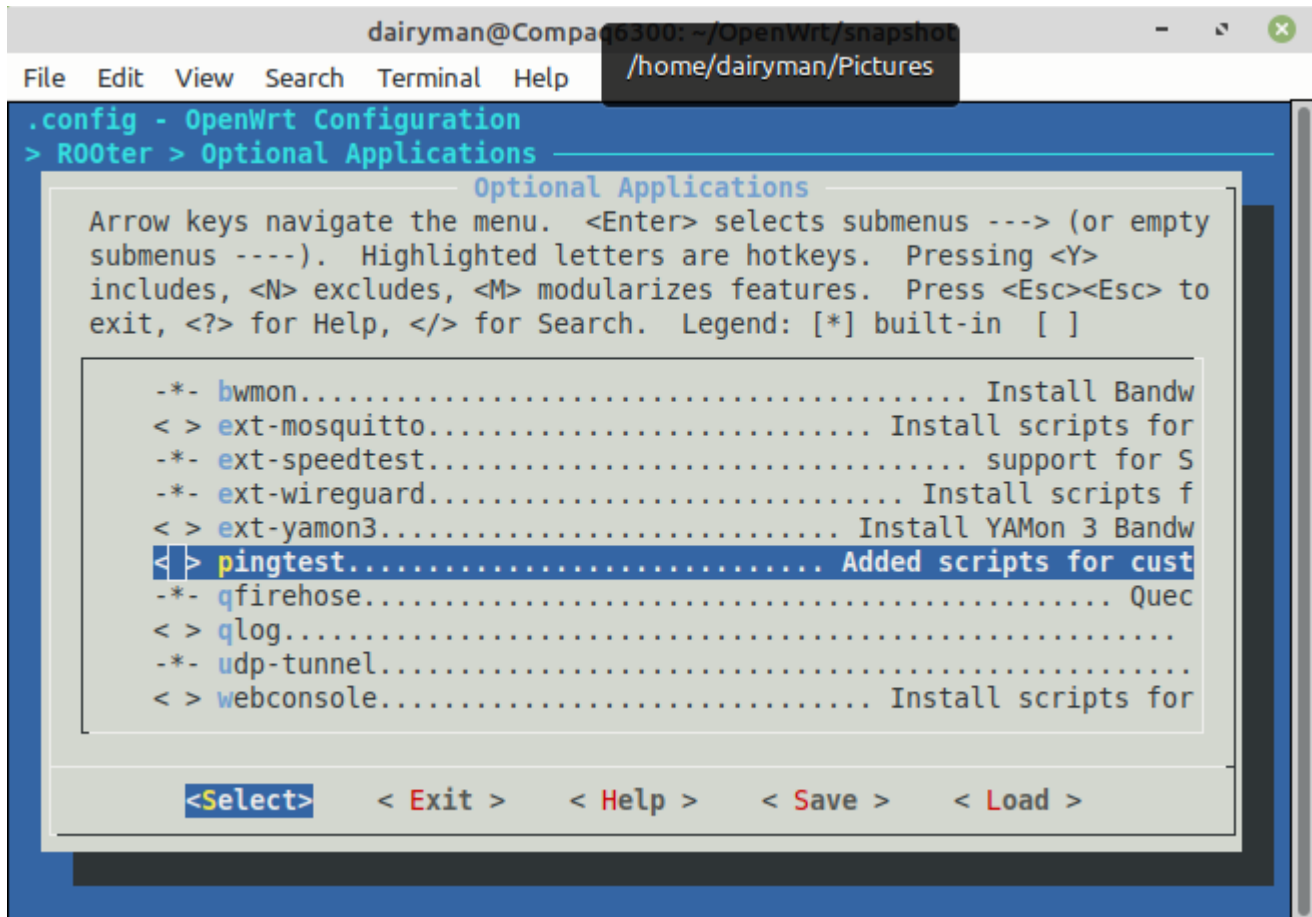
^(-)
< > ext-ssh..... Install scr
< > fscheck..... Install scripts
< > mt1300..... Install script
< > rbm11g..... Install script
< > rbm33g..... Install script
< > rd05a1..... Install script
< > rpi3bp..... Install scripts
< > we826q..... Install scripts
<*> wg1608..... Install script
<|> wrt1900..... Install scripts

<Select> < Exit > < Help > < Save > < Load >
```

Use **Y** to select the extra package if one is available for your router.

Use the left or right arrow key to highlight the **Exit** command at the bottom and press **Enter** to leave this menu.

There also extra ROOter packages like the *Custom Ping Test* that you may wish to add to your image. To see these packages scroll to the **Optional Applications** menu and press **Enter**.



The screenshot shows a terminal window titled "dairyman@Compag6300: ~/OpenWrt/snapshot" with a file explorer overlay showing "/home/dairyman/Pictures". The terminal displays the ".config - OpenWrt Configuration" menu, where the "Optional Applications" option is selected. A detailed help text explains the navigation: "Arrow keys navigate the menu. <Enter> selects submenus ---> (or empty submenus ----). Highlighted letters are hotkeys. Pressing <Y> includes, <N> excludes, <M> modularizes features. Press <Esc><Esc> to exit, <?> for Help, </> for Search. Legend: [\*] built-in [ ]". A list of optional applications follows, with "pingtest" highlighted. The list includes: "bwmon" (Install Bandw), "ext-mosquitto" (Install scripts for), "ext-speedtest" (support for S), "ext-wireguard" (Install scripts f), "ext-yamon3" (Install YAMon 3 Bandw), "pingtest" (Added scripts for cust), "qfirehose" (Quec), "qlog", "udp-tunnel", and "webconsole" (Install scripts for). At the bottom, navigation options are listed: "<Select>", "< Exit >", "< Help >", "< Save >", and "< Load >".

```
dairyman@Compag6300: ~/OpenWrt/snapshot
File Edit View Search Terminal Help /home/dairyman/Pictures

.config - OpenWrt Configuration
> ROOter > Optional Applications

Optional Applications
Arrow keys navigate the menu. <Enter> selects submenus ---> (or empty
submenus ----). Highlighted letters are hotkeys. Pressing <Y>
includes, <N> excludes, <M> modularizes features. Press <Esc><Esc> to
exit, <?> for Help, </> for Search. Legend: [*] built-in [ ]

[*] bwmon..... Install Bandw
< > ext-mosquitto..... Install scripts for
[*] ext-speedtest..... support for S
[*] ext-wireguard..... Install scripts f
< > ext-yamon3..... Install YAMon 3 Bandw
< > pingtest..... Added scripts for cust
[*] qfirehose..... Quec
< > qlog.....
[*] udp-tunnel.....
< > webconsole..... Install scripts for

<Select> < Exit > < Help > < Save > < Load >
```

Use **Y** to select the extra package if one is available for your router.

Use the left or right arrow key to highlight the **Exit** command at the bottom and press **Enter** to leave this menu.

Repeat this procedure in the ROOter menu to return to the main menu.

### **Makeup Of ROOter Packages**

When selecting the main ROOter package you have several choices depending on what features you want. These are :

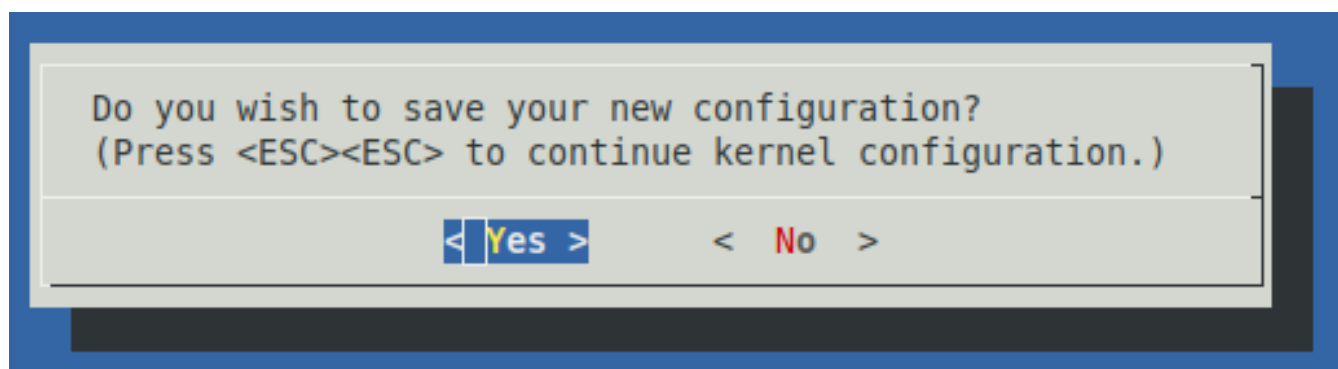


- **ext-rooter-lite** – this is the full package of ROOter features except for MWan3 (Load Balancing). If you are making an image for a router that is not using two modems this is the one to chose. For routers with 16meg or more of flash memory.
- **ext-rooter16** – this is the full package of ROOter features including MWan3 (Load Balancing). This can be used on a router using two modems. For routers with 16meg or more of flash.
- **ext-rooter4** – this package contains all the ROOter modem features but a very limited selection of router level features. This requires the smallest amount of flash memory and will run on any router with 8meg or more of flash.
- **ext-rooter8-vpn** – this contains all of the features of the ext-rooter4 package plus both OpenVpn and Wireguard. It does not include any USB storage features. For routers with 8meg or more of flash.
- **ext-rooterbcm16** – this is for routers using the Broadcom SOC which limited or no wifi support in OpenWrt. This has the same features as ext-rooter16 except for wifi related features such as Hotspot Manager. For routers with 16meg or more of flash.

## Finish the Configuration

With all the required packages selected you can now leave the configuration program. Do this by using the left or right arrow key to highlight the **Exit** command at the bottom and then press **Enter**.

If you have modified the configuration file or have created a new file then you will be asked if you want to save it.



Use the arrow keys to select Yes and press Enter. Your configuration data will be saved in the file named **.config**.

This completes the configuration part of building an image. You have selected the make and model of the router and added the ROOter packages to the configuration file. Now you are ready to compile the image.

## **Compiling the Image**

At this point in a normal OpenWrt build system you would compile the image by entering this in the Terminal

**make V=s**

This command would compile the image and leave it in a subfolder of **/bin**.

Because ROOter is slightly modified from OpenWrt ,and uses different packages than it does, this command will not work.

You will get compile errors about different packages wanting to supply the same files to the image.

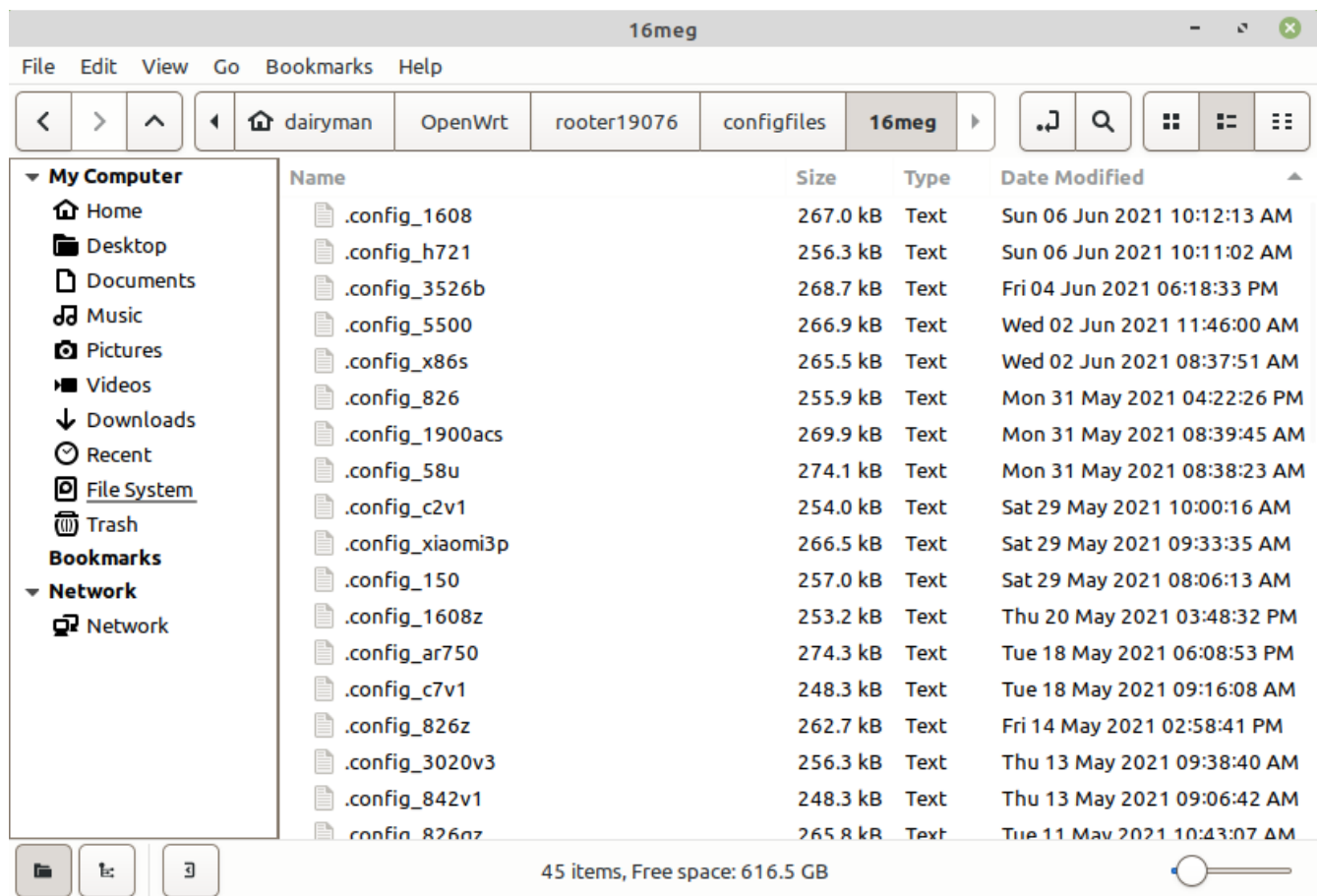
To make it easier to compile a ROOter image that looks and acts exactly like the official ones there is a script that does all the work for you. Once you have set up the script it will build the image, rename it and zip it up ready for use. We will get into the details later.

After you have done the configuration part of the build you are left with a file named **.config** in the main folder. This file is specific to a single router. If you want to make an image for another router this file will be overwritten and you will lose your previous configuration.

In order to avoid this the **.config** files should be renamed and moved to separate folder where they will not be overwritten accidentally.

The policy for the ROOter build script is to rename these files to **.config\_xxx** where **xxx** is an identifier for the router. For example, the **.config** file for the *WG1608* would be renamed to **.config\_1608** and moved to the */configfiles/16meg* folder. This folder would then hold all the **.config** files for all the routers we are making images for.

So the first step after creating a **.config** file is to rename it and move it to the */configfiles/16meg* folder.



This is the `/configfiles/16meg` folder on a build system used to create official ROOter images.

Every router that has an image made for it has a specially named **.config** file in this folder.

## **Build Script**

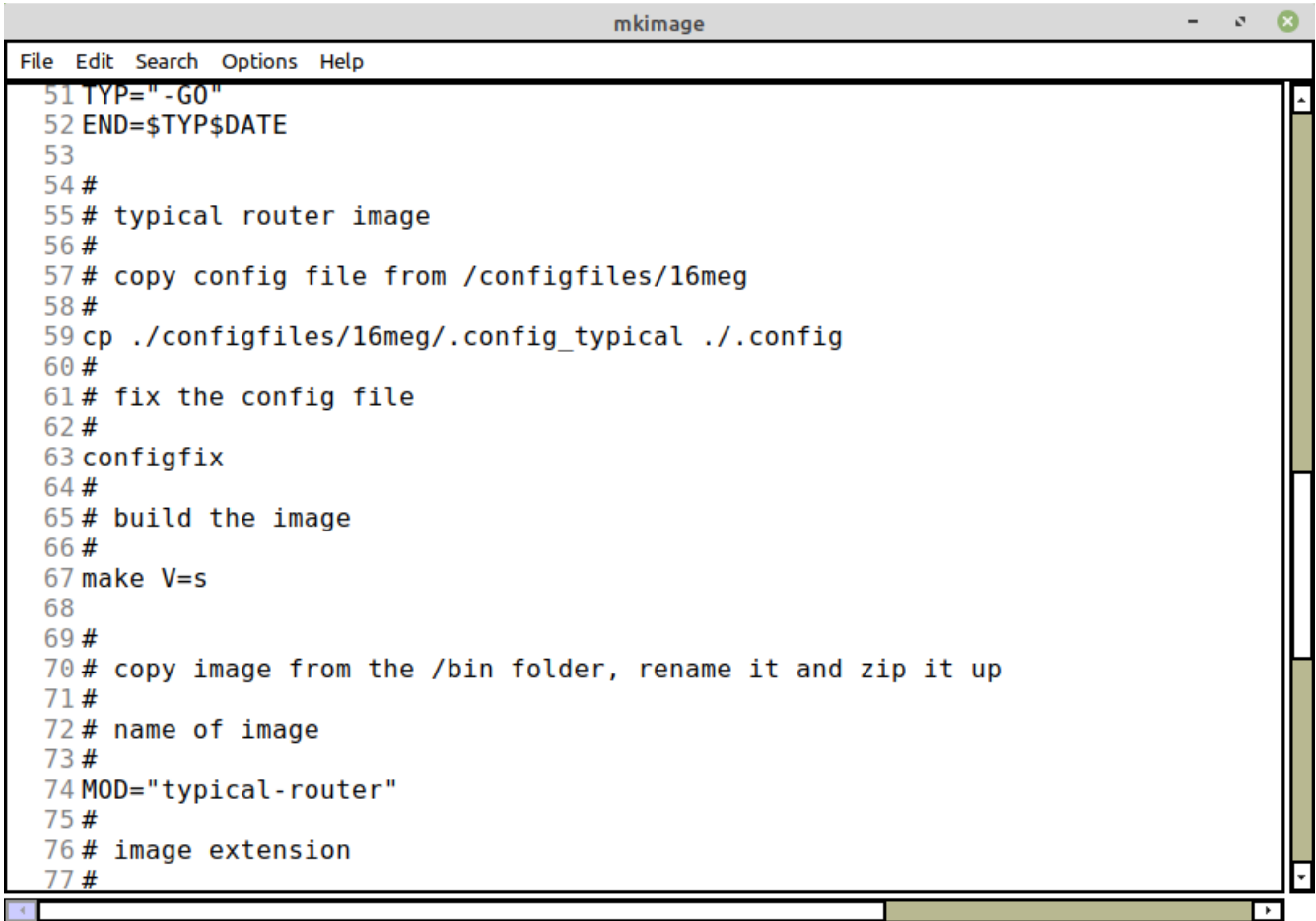
Let us look at the build script to see what it does. This script is named ***mkimage*** and does a number of things to set up and compile the image.

The function at line 10 is used to remove packages that will conflict with other packages added to the image by ROOter. This gets rid of compile time errors about this.

Line 30 sets the date of the image to today's date.

Line 40 adds the images used on the GUI header for the different themes.

The important things start at line 54.

A screenshot of a text editor window titled 'mkimage'. The window has a menu bar with 'File', 'Edit', 'Search', 'Options', and 'Help'. The text content shows a script with line numbers 51 through 77. The script includes comments and commands for building an image, such as copying a config file and running 'make' and 'cp' commands.

```
51 TYP="-G0"
52 END=$TYP$DATE
53
54 #
55 # typical router image
56 #
57 # copy config file from /configfiles/16meg
58 #
59 cp ./configfiles/16meg/.config_typical ./config
60 #
61 # fix the config file
62 #
63 configfix
64 #
65 # build the image
66 #
67 make V=s
68
69 #
70 # copy image from the /bin folder, rename it and zip it up
71 #
72 # name of image
73 #
74 MOD="typical-router"
75 #
76 # image extension
77 #
```

Line 59 is where we copy our renamed **.config** file from */configfiles/16meg* and rename it back to **.config**. You need to edit this line and change **.config\_typical** to the name of the desired file.

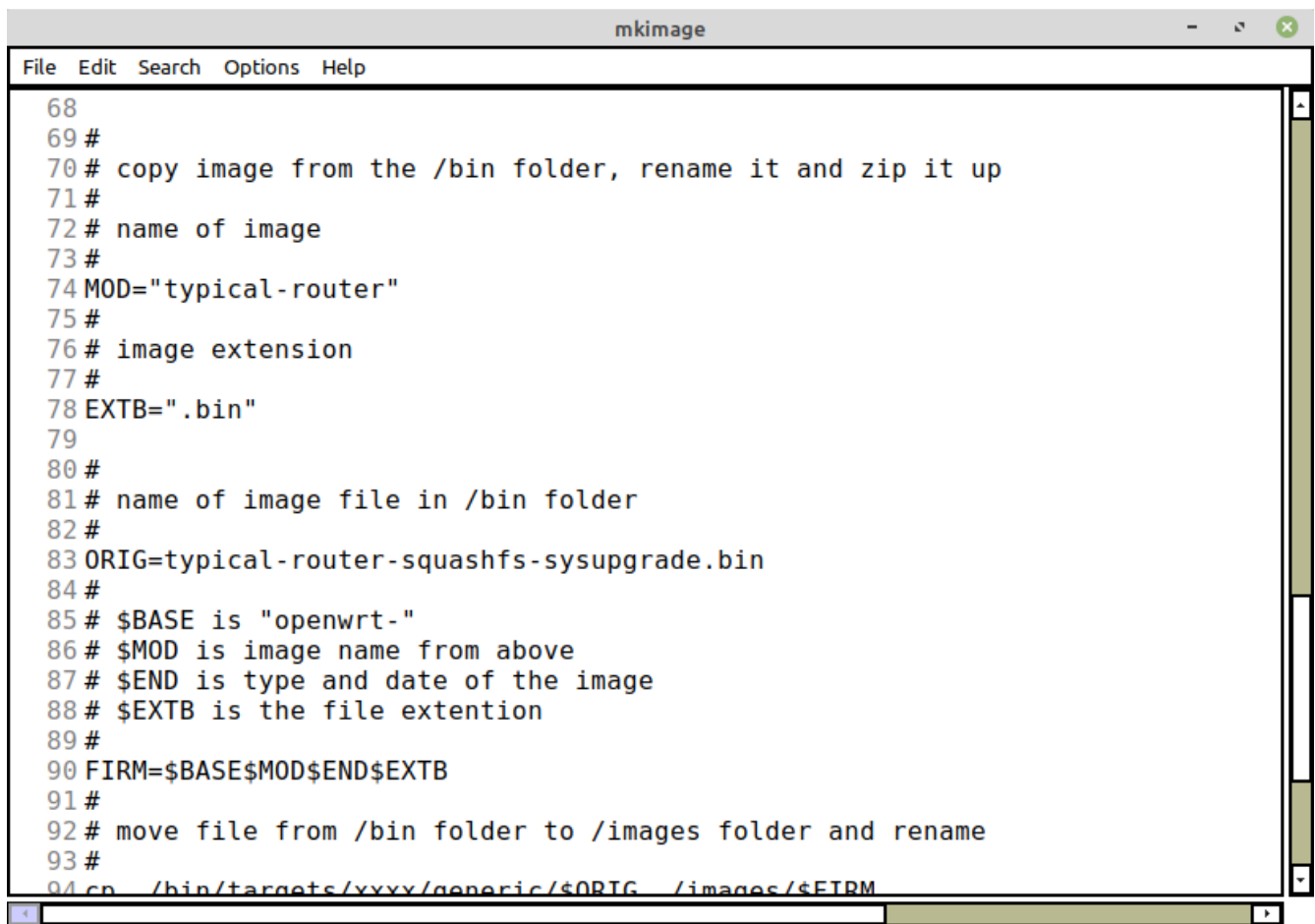
For example, for the WG1608 we would change the line to

**cp ./configfiles/16meg/.config\_1608 ./config**

Line 63 fixes the package conflict issues in the **.config** file and line 67 compiles the image.

When the **make** command finishes running there will be an image in a subfolder of */bin*. To finish our image building we need to copy this into the */images* folder, rename it in the ROOter style and then zip it up.

This what the rest of the script does.

A screenshot of a window titled 'mkimage'. The window has a menu bar with 'File', 'Edit', 'Search', 'Options', and 'Help'. The main area contains a script with the following lines:

```
68
69 #
70 # copy image from the /bin folder, rename it and zip it up
71 #
72 # name of image
73 #
74 MOD="typical-router"
75 #
76 # image extension
77 #
78 EXTB=".bin"
79
80 #
81 # name of image file in /bin folder
82 #
83 ORIG=typical-router-squashfs-sysupgrade.bin
84 #
85 # $BASE is "openwrt-"
86 # $MOD is image name from above
87 # $END is type and date of the image
88 # $EXTB is the file extension
89 #
90 FIRM=$BASE$MOD$END$EXTB
91 #
92 # move file from /bin folder to /images folder and rename
93 #
94 cp /bin/targets/yyyy/generic/$ORIG /images/$FIRM
```

In line 74 we give the image its name. You would replace **typical-router** with something like **ZBT-WG1608** or **GL-AR750**.

Line 78 is the file extension of the image file. Generally this is **bin** but it does vary depending on the router. More on this later.

Line 83 is the name that OpenWrt gives to the image file when it compiles it. Again, more on this later.

Line 90 is the name we want to rename the image file to. This name is made up of 4 parts.

1. BASE is **openwrt-** by default.
2. MOD is the name we gave in line 74.
3. END is the ROOter date and name like GO2021-06-06.
4. EXTB is the extension we defined in line 78.

```
mkimage
File Edit Search Options Help
79
80 #
81 # name of image file in /bin folder
82 #
83 ORIG=typical-router-squashfs-sysupgrade.bin
84 #
85 # $BASE is "openwrt-"
86 # $MOD is image name from above
87 # $END is type and date of the image
88 # $EXTB is the file extension
89 #
90 FIRM=$BASE$MOD$END$EXTB
91 #
92 # move file from /bin folder to /images folder and rename
93 #
94 cp ./bin/targets/xxxx/generic/$ORIG ./images/$FIRM
95 #
96 # switch to /images folder and zip up the image
97 #
98 cd ./images
99 zip $MOD$END.zip $FIRM
100 #
101 # delete unzipped image file and got back to top folder
102 #
103 rm -f $FIRM
104 cd ..
105
```

In line 94 we copy the image file from the correct */bin* subfolder and place it in */images*.

In Line 99 we zip it up in a ROOter style archive.

Then at line 103 we delete the image file to clean up the folder.

To recap the script and how it is used.

1. You need to know the name of the **.config** file in */configfiles/16meg* that is associated with this router and enter it in line 59.
2. You need the router name to be entered into line 74 and the image file extension in line 78.
3. The name of the image file as assigned by OpenWrt goes into line 83
4. The location of the image file in the */bin* folder that goes in line 94.

Once you have made these changes to the script it can be run from the Terminal by entering

**./mkimage**

and the zipped up image will be placed in the */images* folder.

## **Examples**

**1.** Create an image for a WG3526 without Load Balancing but with the Custom Ping Test feature. In the Terminal run

**make menuconfig**

and select

- **Target System** – MediaTek Ralink MIPS
- **Subtarget** – MT7621 based boards
- **Target Profile** – Zbtlink ZBT-WG3526 16M

Then select the following packages.

- **ROOter** → ext-router-lite
- **ROOter** → Optional Applications → pingtest

Exit the configuration program and save the configuration.

Rename the **.config** file to **.config\_3526** and move it to the */configfiles/16meg* folder.

Edit the **mkimage** script at the following lines.

**Line 56** – change **.config\_typical** to **.config\_3526**

```
cp ./configfiles/16meg/.config_3526 ./config
```

**Line 74** – change **typical-router** to **WG3526**

```
MOD="ZBT-WG3526"
```

**Line 83** – change **typical-router-squashfs-sysupgrade.bin** to **openwrt-ramips-mt7621-zbt-wg3526-16M-squashfs-sysupgrade.bin**

```
ORIG="openwrt-ramips-mt7621-zbt-wg3526-16M-squashfs-sysupgrade.bin"
```

**Line 94** – change **xxxx/generic/** to **ramips/mt7621/**

```
cp ./bin/targets/ramips/mt7621/$ORIG ./images/$FIRM
```

Then in the Terminal run

```
./mkimage
```

The image will be compiled and placed in a zip file in */images*.

**2.** Create an image for a Gl.iNet AR150 with Load Balancing, the Custom Ping Test feature and the Web Console. In the Terminal run

```
make menuconfig
```

and select

- **Target System** – Atheros ATH79 (DTS)
- **Subtarget** – Generic
- **Target Profile** – Gl.iNet GL-AR150

Then select the following packages.

- **ROOter** → ext-rooter16
- **ROOter** → Optional Applications → pingtest
- **ROOter** → Optional Applications → webconsole

Exit the configuration program and save the configuration.

Rename the **.config** file to **.config\_150** and move it to the */configfiles/16meg* folder.

Edit the **mkimage** script at the following lines.

**Line 56** – change **.config\_typical** to **.config\_150**

```
cp ./configfiles/16meg/.config_150 ./config
```

**Line 74** – change **typical-router** to **GL-AR150**

```
MOD="GL-AR150"
```



**Line 83** – change **typical-router-squashfs-sysupgrade.bin** to **openwrt-ath79-generic-glinet\_gl-ar150-squashfs-sysupgrade.bin**

***ORIG="openwrt-ath79-generic-glinet\_gl-ar150-squashfs-sysupgrade.bin"***

**Line 94** – change **xxxx/generic/** to **ath79/generic/**

***cp ./bin/targets/ath79/generic/\$ORIG ./images/\$FIRM***

Then in the Terminal run

***./mkimage***

The image will be compiled and placed in a zip file in */images*.

## **Routers with Two Images**

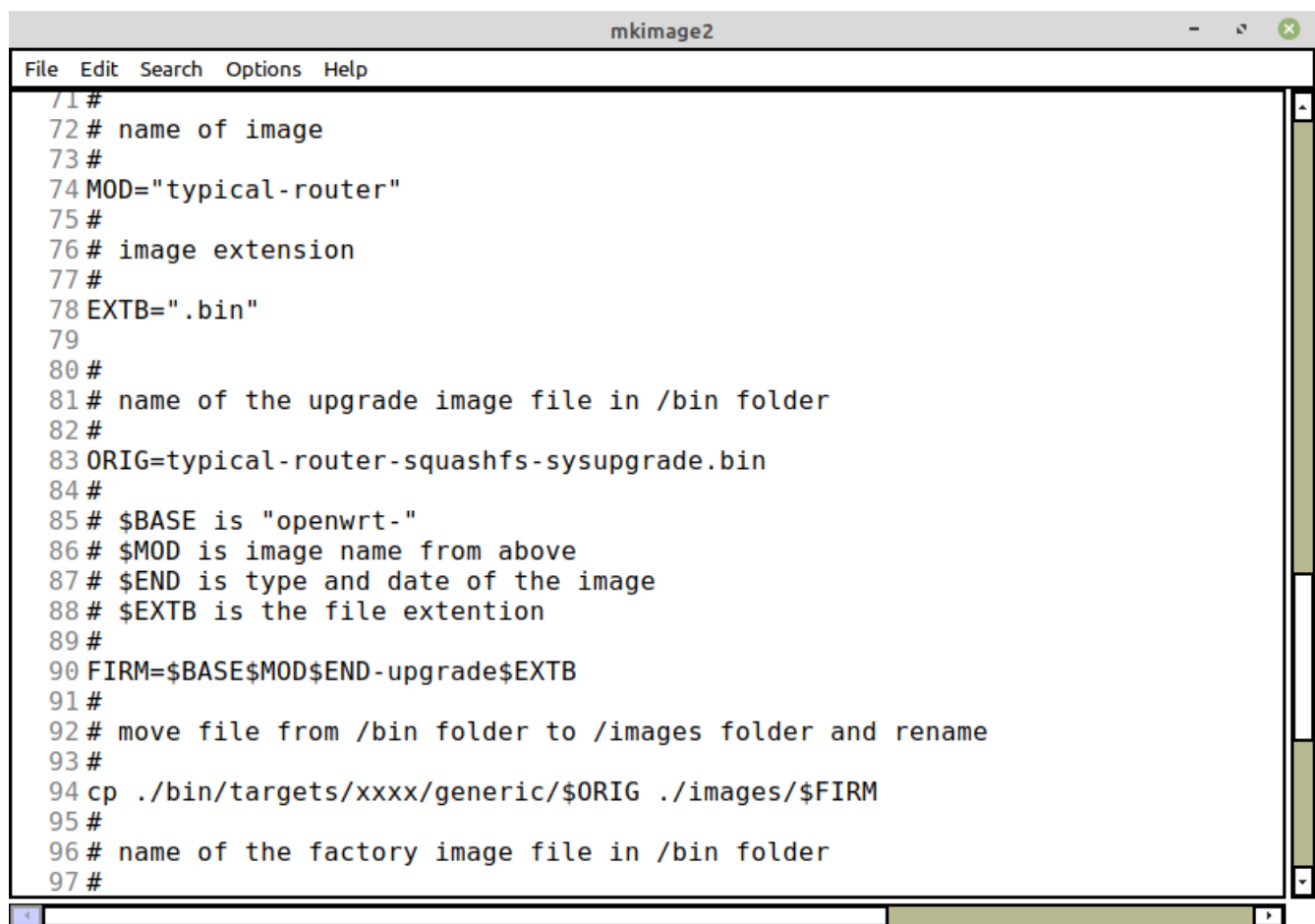
To this point we have looked at building images for routers that use the same image when flashing from the factory firmware and when upgrading an existing ROOter firmware.

A number of routers require a different image for each of these actions so we will look a build script that creates both images.

Creating the **.config** file for these routers is exactly the same as the previous example. When we compile the image OpenWrt will automatically create both types of images if required. The only difference is in taking the correct images from the */bin* subfolder and zipping them up.

The build script for routers requiring two images is **mkimage2** and is very similar to the **mkimage** script.

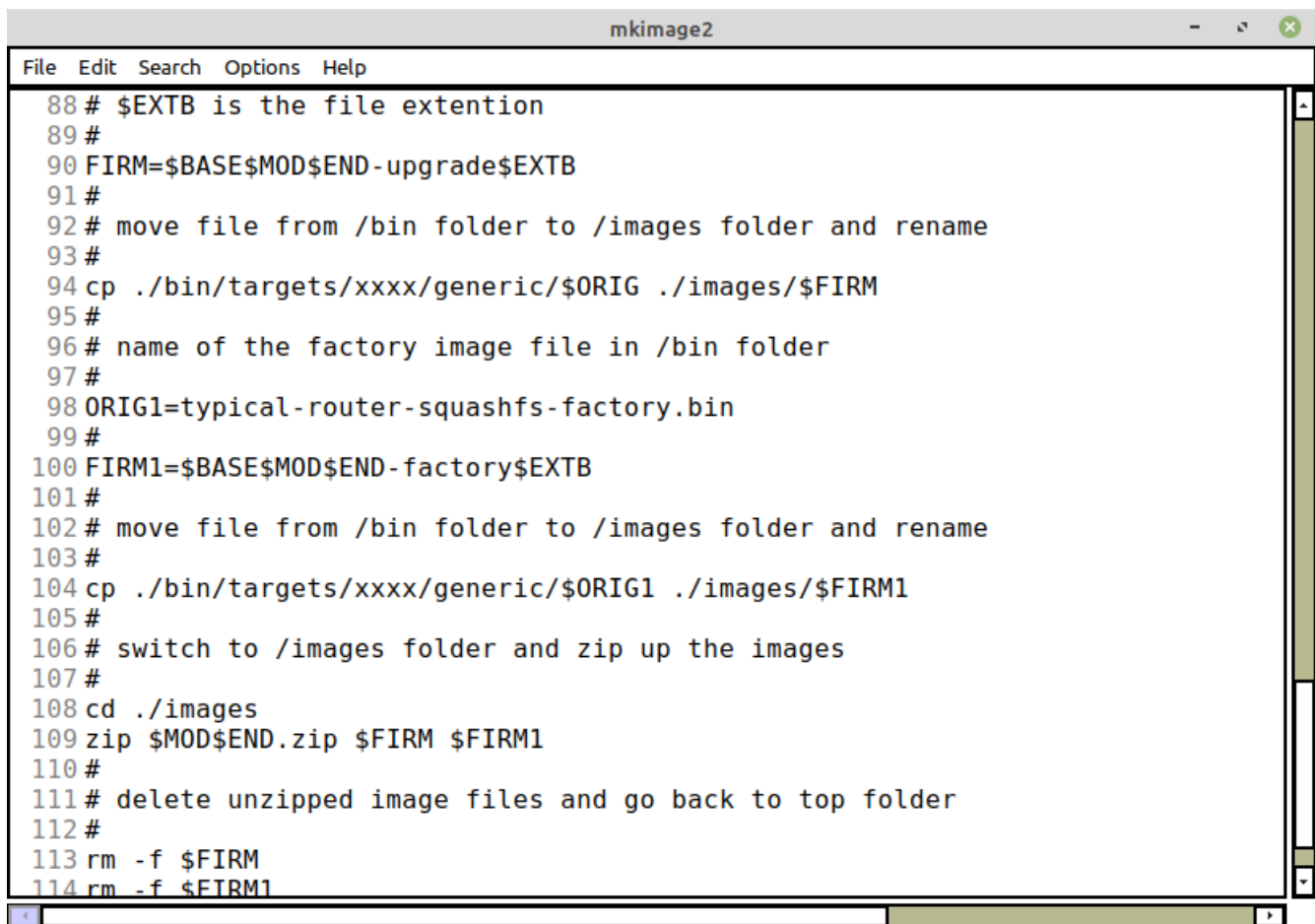
Up to line 78 they are identical and only diverge after that.



```
mkimage2
File Edit Search Options Help
71 #
72 # name of image
73 #
74 MOD="typical-router"
75 #
76 # image extension
77 #
78 EXTB=".bin"
79
80 #
81 # name of the upgrade image file in /bin folder
82 #
83 ORIG=typical-router-squashfs-sysupgrade.bin
84 #
85 # $BASE is "openwrt-"
86 # $MOD is image name from above
87 # $END is type and date of the image
88 # $EXTB is the file extention
89 #
90 FIRM=$BASE$MOD$END-upgrade$EXTB
91 #
92 # move file from /bin folder to /images folder and rename
93 #
94 cp ./bin/targets/xxxx/generic/$ORIG ./images/$FIRM
95 #
96 # name of the factory image file in /bin folder
97 #
```

Line 83 is the name of the image file used to update the router if it is already running ROOter.

Line 90 is the name we wish to give this image. Note the **-upgrade** in the name.

A screenshot of a text editor window titled 'mkimage2'. The window has a menu bar with 'File', 'Edit', 'Search', 'Options', and 'Help'. The main text area contains a shell script with 14 lines. Lines 88-91 are comments defining variables. Line 92 is a comment about moving files. Line 94 is a 'cp' command. Lines 96-99 are comments defining variables. Line 100 is a variable assignment. Line 102 is a comment about moving files. Line 104 is a 'cp' command. Line 106 is a comment about zipping files. Line 108 is a 'cd' command. Line 109 is a 'zip' command. Line 111 is a comment about deleting files. Line 113 is a 'rm' command. Line 114 is a 'rm' command. The script is as follows:

```
88 # $EXTB is the file extention
89 #
90 FIRM=$BASE$MOD$END-upgrade$EXTB
91 #
92 # move file from /bin folder to /images folder and rename
93 #
94 cp ./bin/targets/xxxx/generic/$ORIG ./images/$FIRM
95 #
96 # name of the factory image file in /bin folder
97 #
98 ORIG1=typical-router-squashfs-factory.bin
99 #
100 FIRM1=$BASE$MOD$END-factory$EXTB
101 #
102 # move file from /bin folder to /images folder and rename
103 #
104 cp ./bin/targets/xxxx/generic/$ORIG1 ./images/$FIRM1
105 #
106 # switch to /images folder and zip up the images
107 #
108 cd ./images
109 zip $MOD$END.zip $FIRM $FIRM1
110 #
111 # delete unzipped image files and go back to top folder
112 #
113 rm -f $FIRM
114 rm -f $FIRM1
```

Line 98 is the name of the image that is used to flash from the factory firmware to ROOter.

Line 100 is the name we wish to give to this image. Note the **-factory** in the name.

Line 104 copies the factory image from the */bin* subfolder and places it in */images*.

After this point, both images are zipped up.

To recap the script and how it is used.

5. You need to know the name of the **.config** file in */configfiles/16meg* that is associated with this router and enter it in line 59.
6. You need the router name to be entered into line 74 and the image file extension in line 78.
7. The name of the update image file as assigned by OpenWrt goes into line 83
8. The name we want to call this update image goes into line 90.
9. The location of the update image file in the */bin* folder that goes in line 94.

10. The name of the factory image file as assigned by OpenWrt goes into line 98
11. The name we want to call this update image goes into line 100.
12. The location of the factory image file in the */bin* folder that goes in line 104

Once you have made these changes to the script it can be run from the Terminal by entering

**`./mkimage2`**

## **Examples**

**1.** Create an image for a WRT1900ACS without Load Balancing but with the Custom Ping Test feature. In the Terminal run

**`make menuconfig`**

and select

- **Target System**      – Marvell EBU Armada
- **Subtarget**            – Marvell Armada 37x/38x/XP
- **Target Profile**      – Linksys WRT1900ACS (Shelby)

Then select the following packages.

- **ROOter** → ext-rooter-lite
- **ROOter** → Optional Applications → pingtest

Exit the configuration program and save the configuration.

Rename the **`.config`** file to **`.config_1900acs`** and move it to the */configfiles/16meg* folder.

Edit the **`mkimage`** script at the following lines.

**Line 56** – change **`.config_typical`** to **`.config_1900acs`**

**`cp ./configfiles/16meg/.config_1900acs ./config`**

**Line 74** – change **`typical-router`** to **`WRT1900ACS`**

**`MOD="WRT1900ACS"`**

**Line 83** – change **typical-router-squashfs-sysupgrade.bin** to **openwrt-mvebu-cortexa9-linksys\_wrt1900acs-squashfs-sysupgrade.bin**

***ORIG="openwrt-mvebu-cortexa9-linksys\_wrt1900acs-squashfs-sysupgrade.bin"***

**Line 94** – change **xxxx/generic/** to **ipq40xx/generic/**

***cp ./bin/targets/ipq40xx/generic/\$ORIG ./images/\$FIRM***

**Line 98** – change **typical-router-squashfs-factory.bin** to **openwrt-mvebu-cortexa9-linksys\_wrt1900acs-squashfs-factory.img**

***ORIG1="openwrt-mvebu-cortexa9-linksys\_wrt1900acs-squashfs-factory.img"***

**Line 100** – change **\$EXTB** to **.img**

***FIRM1=\$BASE\$MOD\$END-factory.img***

**Line 104** – change **xxxx/generic/** to **ipq40xx/generic/**

***cp ./bin/targets/ipq40xx/generic/\$ORIG1 ./images/\$FIRM1***

Then in the Terminal run

***./mkimage2***

The image will be compiled and placed in a zip file in */images*.