

**MTK-OpenWrt-3.10.14-SDK**

**Release Notes**

Version: 3.2

Release date: 2015-03-10

© 2014 MediaTek Inc.

This document contains information that is proprietary to MediaTek Inc.

Unauthorized reproduction or disclosure of this information in whole or in part is strictly prohibited.

Specifications are subject to change without notice.

Document Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Revision** | **Date** | **Author** | **Description** |
| 3.0 | 2014.11.10 | Hua Shao | OpenWrt 3.0, with up-to-date kernel & drivers from MTK/Ralink Linux SDK. |
| 3.1 | 2014.11.27 | Dragon Xiong | OpenWrt3.1, support 7628A,7688A, besides update to fix some issue base on 3.0 |
| 3.2 | 2015.03.09 | Dragon Xiong | OpenWrt3.2, add support 7620A,7610E, besides update to fix some issue base on 3.1 |

Table of Contents

[Document Revision History 2](#_Toc413765877)

[Table of Contents 3](#_Toc413765878)

[1 Introduction 6](#_Toc413765879)

[1.1 About OpenWrt 6](#_Toc413765880)

[1.2 About this SDK 6](#_Toc413765881)

[2 Change History 7](#_Toc413765882)

[2.1 V3.0, 20141110 7](#_Toc413765883)

[2.2 V3.1, 20141127 7](#_Toc413765884)

[2.3 V3.2, 20150310 7](#_Toc413765885)

[3 SDK Files 9](#_Toc413765886)

[4 Build the SDK 10](#_Toc413765887)

[4.1 Setup Build Environment 10](#_Toc413765888)

[4.2 Check Build Dependency 10](#_Toc413765889)

[4.3 SDK root folder 10](#_Toc413765890)

[4.4 Config 11](#_Toc413765891)

[4.4.1 Config OpenWrt 11](#_Toc413765892)

[4.4.2 Config Linux Kernel. 11](#_Toc413765893)

[4.5 Build 12](#_Toc413765894)

[4.6 Install Firmware 13](#_Toc413765895)

[5 Web Interface 15](#_Toc413765896)

[5.1 LuCI 15](#_Toc413765897)

[5.1.1 Install 15](#_Toc413765898)

[5.1.2 Config & Build 15](#_Toc413765899)

[5.1.3 Access 16](#_Toc413765900)

[5.2 LuCI-mtk 16](#_Toc413765901)

[5.2.1 Install 16](#_Toc413765902)

[5.2.2 Configure wifi via luci-mtk 17](#_Toc413765903)

[1 Wireless configuration via UCI 21](#_Toc413765904)

[1.1 Basic idea 21](#_Toc413765905)

[1.2 Examples 21](#_Toc413765906)

[1.2.1 SSID 21](#_Toc413765907)

[1.2.2 Encryption 22](#_Toc413765908)

[1.2.3 Key 23](#_Toc413765909)

[1.2.4 Add new SSID (When multi-SSID is enabled) 23](#_Toc413765910)

[1.2.5 Remove SSID 23](#_Toc413765911)

[1.2.6 Wireless Mode 23](#_Toc413765912)

[1.2.7 Radio On/Off 24](#_Toc413765913)

[1.2.8 Channel 24](#_Toc413765914)

[1.2.9 Operating Mode 24](#_Toc413765915)

[1.2.10 Channel Band Width 24](#_Toc413765916)

[1.2.11 Guard Interval 24](#_Toc413765917)

[1.2.12 MCS 25](#_Toc413765918)

[1.2.13 Reverse Direction Grant 25](#_Toc413765919)

[1.2.14 Space Time Block Coding (STBC) 25](#_Toc413765920)

[1.2.15 Aggregation MSDU (A-MSDU) 25](#_Toc413765921)

[1.2.16 AP Power Saving (APSDCapable) 25](#_Toc413765922)

[1.2.17 WMM 25](#_Toc413765923)

[1.2.18 Decline BA Request 26](#_Toc413765924)

[1.2.19 HT LDPC 26](#_Toc413765925)

[1.2.20 BG Protection Mode 26](#_Toc413765926)

[1.2.21 VHT BW Signaling 26](#_Toc413765927)

[1.2.22 Beacon Interval 26](#_Toc413765928)

[1.2.23 Data Beacon Rate (DTIM) 26](#_Toc413765929)

[1.2.24 Fragment Threshold 26](#_Toc413765930)

[1.2.25 RTS Threshold 27](#_Toc413765931)

[1.2.26 TX Power 27](#_Toc413765932)

[1.2.27 Short Preamble 27](#_Toc413765933)

[1.2.28 Short Slot 27](#_Toc413765934)

[1.2.29 Tx Burst 27](#_Toc413765935)

[1.2.30 Pkt\_Aggregate 27](#_Toc413765936)

[1.2.31 IEEE 802.11H Support 27](#_Toc413765937)

[1.2.32 Country Code 27](#_Toc413765938)

[1.2.33 2.4G Country Region 28](#_Toc413765939)

[1.2.34 5G Country Region 28](#_Toc413765940)

[2 MTK/Ralink Property Packages 29](#_Toc413765941)

[2.1 Applications 29](#_Toc413765942)

[2.1.1 ated 29](#_Toc413765943)

[2.1.2 btnd 29](#_Toc413765944)

[2.1.3 eth\_mac 29](#_Toc413765945)

[2.1.4 ethstt 29](#_Toc413765946)

[2.1.5 gpio 30](#_Toc413765947)

[2.1.6 nvram 30](#_Toc413765948)

[2.1.7 mii\_mgr 31](#_Toc413765949)

[2.1.8 reg 31](#_Toc413765950)

[2.1.9 switch 31](#_Toc413765951)

[2.1.10 uci2dat 32](#_Toc413765952)

[2.1.11 watchdog 33](#_Toc413765953)

[2.2 Drivers 33](#_Toc413765954)

[2.2.1 MT7620 34](#_Toc413765955)

[2.2.2 MT7610e 34](#_Toc413765956)

[2.2.3 MT7628/MT7688 34](#_Toc413765957)

[2.2.4 MT7603e 34](#_Toc413765958)

[2.2.5 MT76x2e 34](#_Toc413765959)

[3 Patches 36](#_Toc413765960)

# Introduction

## About OpenWrt

OpenWrt (<http://www.openwrt.org/>) is a linux distribution primarily used on embedded devices to route network traffic. The main components are the Linux kernel, uClibc, busybox, and OpenWrt framework utilities. All components have been optimized for size, to be small enough for fitting into the limited storage and memory available in the routers.

## About this SDK

This SDK is a MTK customized OpenWrt project.

To provide better compatibility and better stability, some OpenWrt drivers were replaced with MTK drivers, such as Ethernet, USB, WiFi, SD Card, etc.

Brief Summary about this SDK:

* OpenWrt framework: Barrier Breaker
* Linux Kernel: 3.10.14
* Toolchain: toolchain-mipsel\_24kec+dsp\_gcc-4.8-linaro\_uClibc-0.9.33.2
* MTK Linux SDK base: linux-3.10.14
* Supported SoC Chips: MT7620, MT7621, MT7628, MT7688
* Supported WiFi Chips: MT7603e, MT7602e, MT7612e, MT7628, MT7620, MT7610e

# Change History

## V3.0, 20141110

Feature:

* SoC chip support: MT7621, MT7620, MT7628
* WiFi chip support: MT7603e. MT7602e, MT7612e, MT7620, MT7610e, MT7628
* Ethernet driver Ready
* Flash driver ready
* PCI-e driver ready
* USB driver ready
* SDXC driver ready
* Ralink apps (8021xd, ated, btnd, gpio, nvram, mii\_mgr, reg, switch, uci2dat, watchdog) ready
* Support luci-mtk web UI.

## V3.1, 20141127

Feature:

* Add support for MT7628A,7688A SoC chip
* Add support for MT7628A, 7688A WiFi Chip
* Support hardware NAT for MT7621
* Support ntfs/extfat fs auto mount

Update:

* Update 7603 driver to fix WMM and RTS/CTS issue
* Update 7612e package to support SingleSKU
* Init 802.1xd when WPA/WPA2 enterprise encryption
* Fix some web issues

Note: Compile 7688a image, please select Subtarget (MT7628 based boards).

## V3.2, 20150310

Feature:

* SoC chip support: MT7621, MT7620, MT7628, MT7688
* WiFi chip support: MT7603e. MT7602e, MT7612e, MT7620, MT7610e, MT7628
* Add source code for MTK APSoC U-boot
* Add 76x2e led support.

Update:

* WiFi: Bug fixes and performance tunning.
* Rewrite nvram and libnvram.
* Adjust default configuration to reduce CPU usage.
* Mt7620: support wifi hwnat.
* Update 7628 driver to lsdk V4.0.1.3\_DPA\_20150216
* Update Nand flash driver to P4 115775 for solving some nand flash issues
* Update 76x2e default config to improve througput and so on

# SDK Files

* **MTK-OpenWrt-3.10.14-SDK-Release Notes.docx**
  + This document.
* **mtksdk-openwrt-3.10.14-{version}-{date}-{tag}.tar.bz2**
  + SDK
* **openwrt-ramips-{chip-id}-squashfs-sysupgrade.bin**
  + Pre-build software

# Build the SDK

## Setup Build Environment

To build this SDK, you should have a linux server (linux 2.6.x or later) as the build host.

The default build will take up to 6 GB disk space. Make sure you have enough space to hold it.

Prepare the source project:

**tar xjvf mtksdk-openwrt-3.10.14-{date}-{tag}.mini.tar.bz2 -C /path/to/your/workspace**

## Check Build Dependency

In the first build, OpenWrt will check your build environment. If it complains that some library or software is missing, you should install them first, eg:

**yum install svn**

**yum install wget**

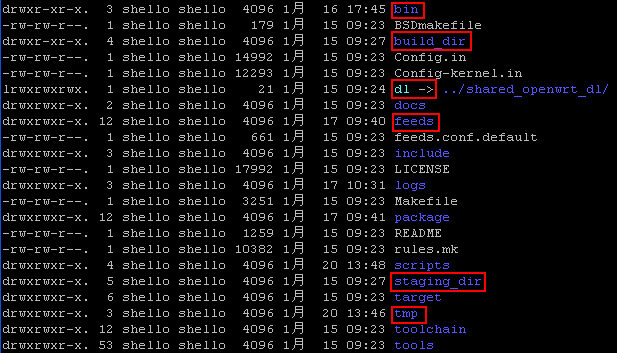
**yum install ncurses-devel**

**yum install zlib-devel**

**…..**

## SDK root folder

This is what the SDK root folder looks like (Those folder names surrounded with red line are auto generated during build).



## Config

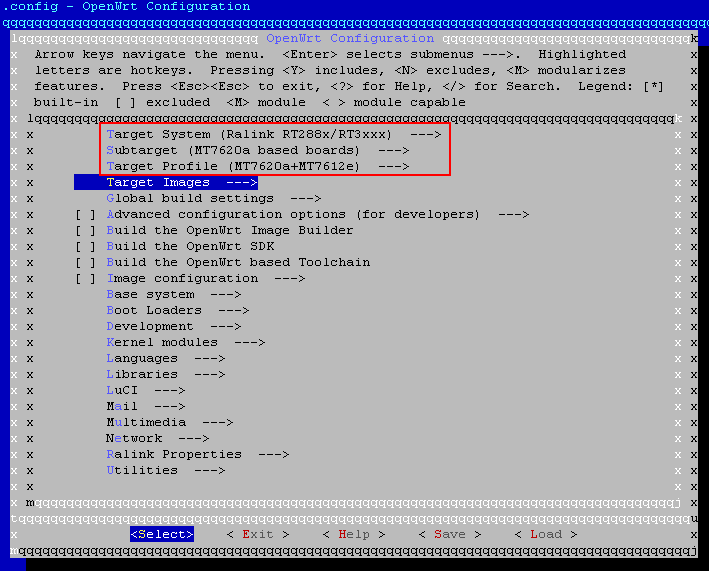
### Config OpenWrt

Under SDK root folder, call:

**make menuconfig**

Then specify you configuration. For a default build, you need at least 3 items:

* Target System (Ralink Platform)
* Subtarget (Ralink SoC chip series)
* Target Profile (A specific model name)



After menuconfig done, you configuration will be saved in /SDK root/.config

Note: In OpenWrt3.2, WiFi chips is not defined in profile. You can choose wifi drivers by yourself based on your HW.

### Config Linux Kernel.

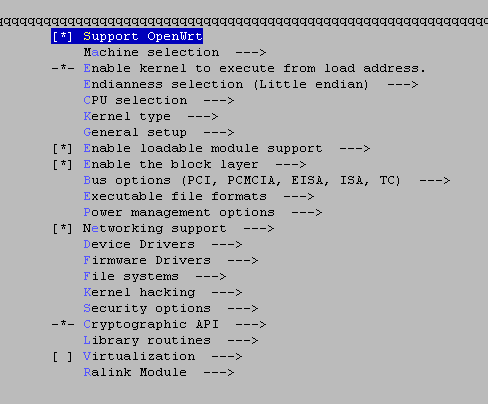
We provide default kernel configuration. you can find it at target/linux/ramips/mt76xx/config-3.10.14.

If that does not meet your needs, you can configure the kernel by yourself.

Under SDK root folder, call:

**make kernel\_menuconfig**

Then you will see the classic kernel configuration menu like this:



## Build

Under SDK root folder, call:

**make**

Or

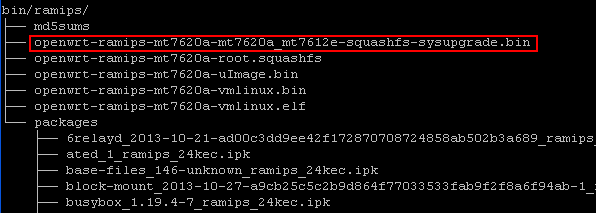
**make V=s** # this will produce verbose log

During build, the SDK will download many source packages from Internet. So, make sure your build host can access the open Internet.

The first build will take hours, please be patient. After first build, your build will be ready in minutes.

If anything goes wrong during building, use “make V=s” to see what happened.

If everything is OK, the target image will be generated under “bin/ramips”.

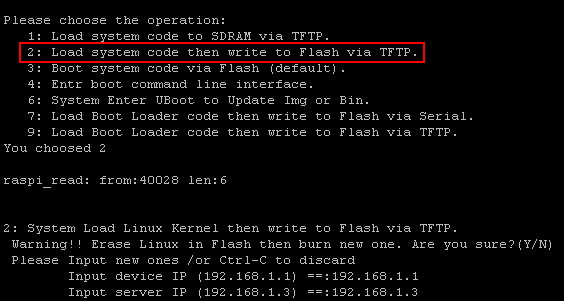


Note: 3.10.14 kernel should use MTK’s kernel, not the original linux kernel.

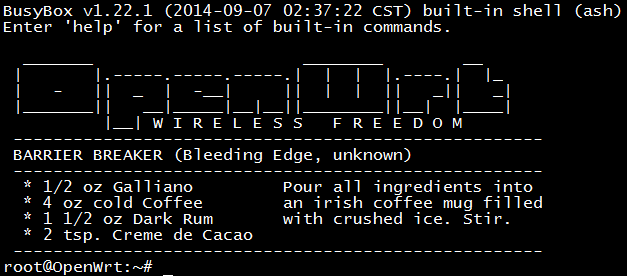
## Install Firmware

OpenWrt firmware can be flashed into the target board using MTK bootloader option 2.

Note: Option 1 won’t work, because the image does not support initram mechanism.



After system reboot, you will see OpenWrt running.



# Web Interface

OpenWrt does not build the web interface by default. Web interface is provided as a 3rd party package. Such as LuCI and XWRT.

## LuCI

### Install

Under SDK root folder, call:

**scrips/feeds update -a**

**scripts/feeds install luci**

The LuCI package will be installed into SDK.

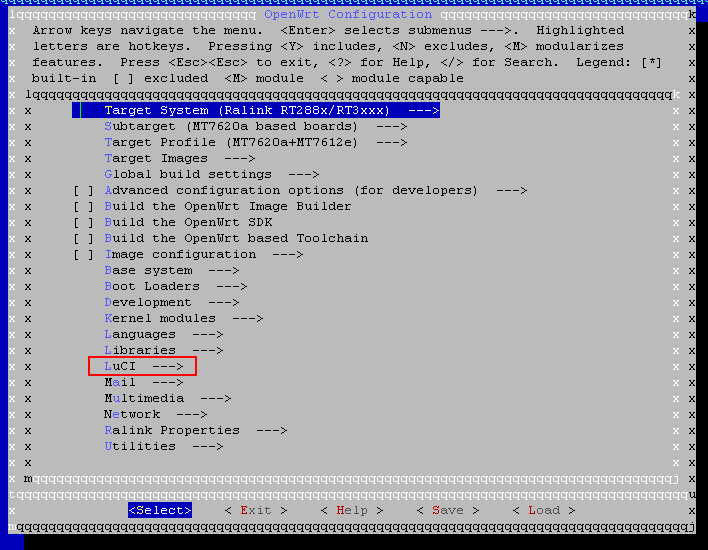
### Config & Build

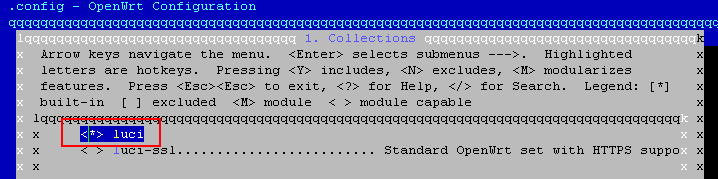
After installing LuCI, a submenu called “LuCI” will show up in “menuconfig”.

LuCI is not selected by default, choose “\*” in “LuCI”->”Collection”->”luci" to enable LuCI by default.Then:

**make V=s**

You will see that LuCI get build along with the SDK.



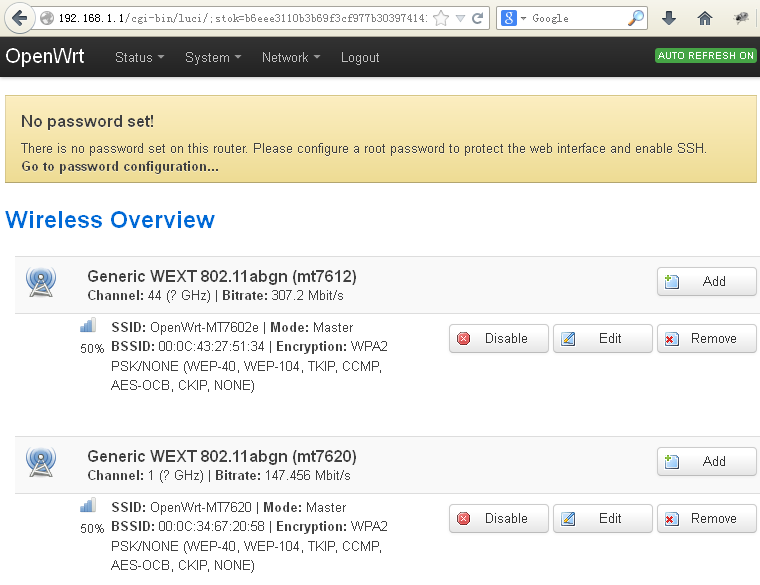


### Access

By default, You can access the web interface at <http://192.168.1.1/>.

**Account = “admin”**

**Password = “admin”**



## LuCI-mtk

### Install

We provide a customized LuCI UI, called luci-mtk.

To install luci-mtk, you should remove the official luci ui first.

If you haven’t installed official luci yet, you can skip this step.

**make package/luci/clean**

**scripts/feeds uninstall luci**

**rm -rf tmp**

**rm .config\***

Then,

**make menuconfig**

In “ Ralink Properties” ->” UI” ->” luci-mtk”, check it as “\*”.

cid:image002.png@01CF5D4B.8F725980

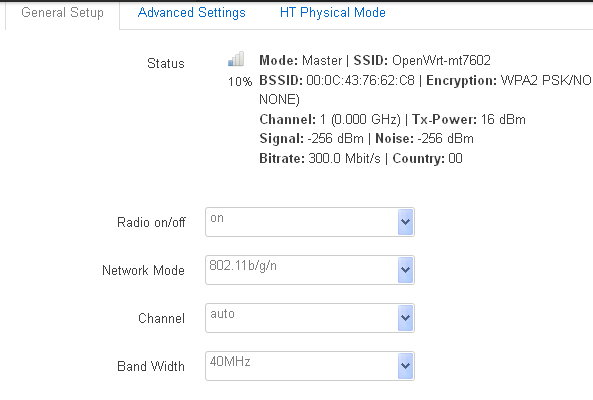
Then luci-mtk will be installed into your image.

We changed the OpenWrt title to make sure you installed the right luci ui.

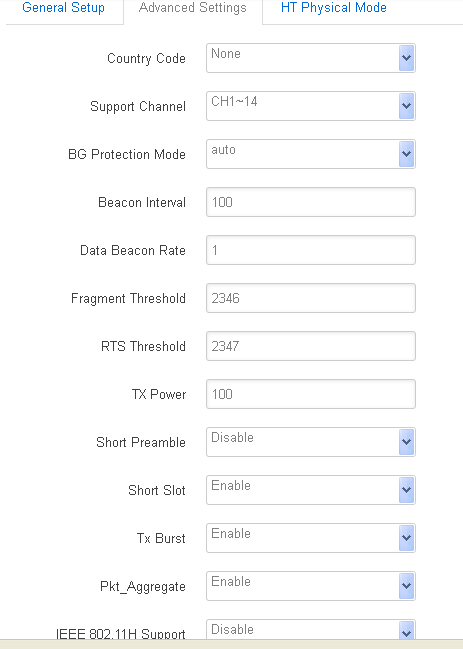
cid:image003.png@01CF5D4B.8F725980

### Configure wifi via luci-mtk

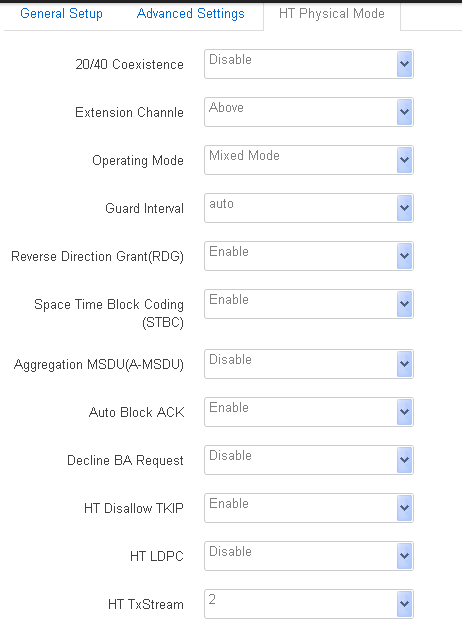
General setup



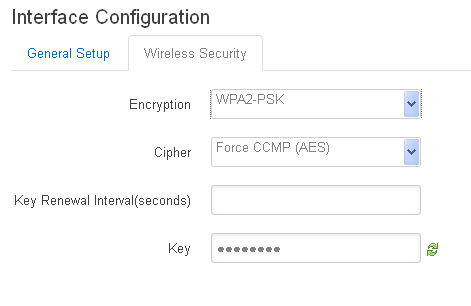
Advanced Settings



HT Physical Mode



Security settings



# Wireless configuration via UCI

## Basic idea

There are already 2 ways to configure MTK wireless drivers.

1. Using iwpriv command. (eg: *iwpriv ra0 set SSID=myrouter*)
2. Edit the profile of the driver. (located at */etc/Wireless/chipname/chipname.dat*)

Here we introduce the 3rd way, which is for the convenience of LuCI development.

To use this feature, you should enable **uci2dat** first. You can find this application at:

*Menuconfig -> Ralink Properties -> Applications -> uci2dat*



Then you can configure MTK wireless drivers via uci commands, like this:

**uci set wireless.chipname.option1=value1**

**uci set wireless.chipname.option2=value2**

**……**

**uci commit**

**wifi down**

**wifi up**

## Examples

In your script, you should replace “chipname” with the right name of your wireless chip, (like mt7620, mt7612, mt7602, mt7610 etc).

### SSID

This is a little tricky, read the example carefully.

**uci set wireless.@wifi-iface[n].ssid=newssid**

N is the index of the interface you want to change. You can check the interface index by :

**uci show wireless**

You may see something like this:

[root@OpenWrt]uci show wireless

wireless.mt7612=wifi-device

wireless.mt7612.type=mt7612

wireless.mt7612.vendor=ralink

wireless.mt7612.channel=0

wireless.mt7612.autoch=2

wireless.@wifi-iface[0]=wifi-iface

wireless.@wifi-iface[0].device=mt7612

wireless.@wifi-iface[0].ifname=rai0

wireless.@wifi-iface[0].network=lan

wireless.@wifi-iface[0].mode=ap

wireless.@wifi-iface[0].ssid=OpenWrt-mt7612

wireless.@wifi-iface[0].encryption=psk2

wireless.@wifi-iface[0].key=12345678

wireless.mt7620=wifi-device

wireless.mt7620.type=mt7620

wireless.mt7620.vendor=ralink

wireless.mt7620.channel=0

wireless.mt7620.autoch=2

wireless.@wifi-iface[1]=wifi-iface

wireless.@wifi-iface[1].device=mt7620

wireless.@wifi-iface[1].ifname=ra0

wireless.@wifi-iface[1].network=lan

wireless.@wifi-iface[1].mode=ap

wireless.@wifi-iface[1].ssid=OpenWrt-mt7620

wireless.@wifi-iface[1].encryption=psk2

wireless.@wifi-iface[1].key=12345678

Then you have 2 WiFi interfaces, one has SSID “OpenWrt-mt7620” and the other has SSID “OpenWrt-mt7612”.

If you want to change “OpenWrt-mt7620” to “MyNew7620”, you should call:

**uci set wireless.@wifi-iface[1].ssid=MyNew7620** # “1” is the index of “OpenWrt-mt7620”

### Encryption

Read the example in “SSID” section first, then you can:

**uci set wireless.@wifi-iface[n].encryption=x**

x could be:

|  |  |
| --- | --- |
| x | encryption |
| psk-mixed | WPAPSKWPA2PSK |
| psk2 | WPA2PSK |
| psk | WPAPSK |
| wpa-mixed | WPAWPA2 |
| wpa2 | WPA2 |
| wpa | WPA |
| wep-open | OPEN |
| wep-shared | SHARED |
| open | OPEN |

### Key

Read the example in “SSID” section first, then you can:

**uci set wireless.@wifi-iface[n].key=x**

### Add new SSID (When multi-SSID is enabled)

Read the example in “SSID” section first, then you can:

**uci add wireless wifi-iface**

**uci set wireless.@wifi-iface[n].device=chipname**

**uci set wireless.@wifi-iface[n].ifname=ra1**

**uci set wireless.@wifi-iface[n].network=lan**

**uci set wireless.@wifi-iface[n].mode=ap**

**uci set wireless.@wifi-iface[n].ssid=newssid**

**uci set wireless.@wifi-iface[n].encryption=psk2**

**uci set wireless.@wifi-iface[n].key=11111111**

N is the new index, which is current index increased by 1.

### Remove SSID

Read the example in “SSID” section first, then you can:

**uci del wireless.@wifi-iface[n]**

N is the index which you want to remove.

### Wireless Mode

**uci set wireless.chipname.wifimode=n**

n could be:

0: Legacy 11b/g mixed

1: Legacy 11B only

2: Legacy 11A only

3: Legacy 11a/b/g mixed

4: Legacy 11G only

5: 11ABGN mixed

6: 11N only

7: 11GN mixed

8: 11AN mixed

9: 11BGN mixed

10: 11AGN mixed

11: 11N only in 5G band only

14: 11A/AN/AC mixed 5G band only (Only 11AC chipset support)

15:11 AN/AC mixed 5G band only (Only 11AC chipset support)

### Radio On/Off

**uci set wireless.chipname.radio=n**

n could be:

0: Disable

1: Enable

### Channel

1. Fixed channel:

**uci set wireless.chipname.autoch=0**

**uci set wireless.chipname.channel=n # n is the channel number.**

1. Auto channel:

**uci set wireless.chipname.channel=0**

**uci set wireless.chipname.autoch=1 # or 2, different algorithm.**

1. Auto channel with channels skipped

**uci set wireless.chipname.channel=0**

**uci set wireless.chipname.autoch=1 # or 2, different algorithm.**

**uci set wireless.chipname.autoch\_skip=”1;2;3;4”**

### Operating Mode

**uci set wireless.chipname.ht\_opmode=n**

n could be:

0: Mixed Mode

1: Green Mode

### Channel Band Width

**uci set wireless.chipname.bw=n**

n could be:

0: 20MHz

1: 40MHz

2: 80MHz (5G only)

### Guard Interval

**uci set wireless.chipname.ht\_gi=n # 2.4G**

n could be:

0: long GI

1: Auto GI

**uci set wireless.chipname.vht\_sgi=n # 5G**

n could be:

0: Disable

1: Enable

### MCS

**uci set wireless.chipname.ht\_mcs=0;....;1;0;**

bitwise data, bit index equals to the MCS index.

### Reverse Direction Grant

**uci set wireless.chipname.ht\_rdg=n**

n could be:

0: Disable

1: Enable

### Space Time Block Coding (STBC)

**uci set wireless.chipname.ht\_stbc=n #2.4G**

n could be:

0: Disable

1: Enable

**uci set wireless.chipname.vht\_stbc=n #5G**

n could be:

0: Disable

1: Enable

### Aggregation MSDU (A-MSDU)

**uci set wireless.chipname.ht\_amsdu=n**

n could be:

0: Disable

1: Enable

### AP Power Saving (APSDCapable)

**uci set wireless.chipname.apsd=n**

n could be:

0: Disable

1: Enable

### WMM

**uci set wireless.chipname.wmm=n**

n could be:

0: Disable

1: Enable

### Decline BA Request

**uci set wireless.chipname.ht\_badec=n**

n could be:

0: Disable

1: Enable

### HT LDPC

**uci set wireless.chipname.ht\_ldpc=n # 2.4G**

n could be:

0: Disable

1: Enable

**uci set wireless.chipname.vht\_ldpc=n # 5G**

n could be:

0: Disable

1: Enable

### BG Protection Mode

**uci set wireless.chipname.bgprotect=n**

n could be:

0: Auto

1: On

2: Off

### VHT BW Signaling

**uci set wireless.chipname.vht\_bw\_sig=n**

n could be:

0: Disable

1: Static

2: Dynamic

### Beacon Interval

**uci set wireless.chipname.beacon=n**

n could be: 20~999

### Data Beacon Rate (DTIM)

**uci set wireless.chipname.dtim=n**

n could be: 1~255

### Fragment Threshold

**uci set wireless.chipname.fragthres=n**

n could be: 255~2346

### RTS Threshold

**uci set wireless.chipname.rtsthres=n**

n could be: 1~2347

### TX Power

**uci set wireless.chipname.txpower=n**

n could be: 1~100

### Short Preamble

**uci set wireless.chipname.txpreamble=n**

n could be:

0: Disable

1: Enable

### Short Slot

**uci set wireless.chipname.shortslot=n**

n could be:

0: Disable

1: Enable

### Tx Burst

**uci set wireless.chipname.txburst=n**

n could be:

0: Disable

1: Enable

### Pkt\_Aggregate

**uci set wireless.chipname.pktaggre=n**

n could be:

0: Disable

1: Enable

### IEEE 802.11H Support

**uci set wireless.chipname.pktaggre=n**

n could be:

0: Disable

1: Enable

### Country Code

**uci set wireless.chipname.country=str**

str could be: **”US”, ”JP” ,”FR” ,”TW”, ”IE” ,”HK” ,”NONE”**

### 2.4G Country Region

**uci set wireless.chipname.region=n**

|  |  |
| --- | --- |
| Region | Channels |
| 0 | 1-11 |
| 1 | 1-13 |
| 2 | 10-11 |
| 3 | 10-13 |
| 4 | 14 |
| 5 | 1-14 |
| 6 | 3-9 |
| 7 | 5-13 |
| 31 | 1-14 |
| 32 | 1-11 active scan, 12 and 13 passive scan |
| 33 | 1-14 all active scan, 14 b mode only |

### 5G Country Region

**uci set wireless.chipname.aregion=n**

|  |  |
| --- | --- |
| Region | Channels |
| 0 | 36, 40, 44, 48, 52, 56, 60, 64, 149, 153, 157, 161, 165 |
| 1 | 36, 40, 44, 48, 52, 56, 60, 64, 100, 104, 108, 112, 116, 120, 124, 128, 132, 136, 140 |
| 2 | 36, 40, 44, 48, 52, 56, 60, 64 |
| 3 | 52, 56, 60, 64, 149, 153, 157, 161 |
| 4 | 149, 153, 157, 161, 165 |
| 5 | 149, 153, 157, 161 |
| 6 | 36, 40, 44, 48 |
| 7 | 36, 40, 44, 48, 52, 56, 60, 64, 100, 104, 108, 112, 116, 120, 124, 128, 132, 136, 140, 149, 153, 157, 161, 165 |
| 8 | 52, 56, 60, 64 |
| 9 | 36, 40, 44, 48, 52, 56, 60, 64, 100, 104, 108, 112, 116, 132, 136, 140, 149, 153, 157, 161, 165 |
| 10 | 36, 40, 44, 48, 149, 153, 157, 161, 165 |
| 11 | 36, 40, 44, 48, 52, 56, 60, 64, 100, 104, 108, 112, 116, 120, 149, 153, 157, 161 |

# MTK/Ralink Property Packages

Here are packages located under **{SDKRoot}/package/ralink**.

Here is a brief introduction to them.

## Applications

### ated

ATE daemon. For factory test.

You should enable the “ATE/QA” option in the driver configuration. (Check “make menuocnfig” -> “ralink properties” -> “drivers” -> “mtxxxx”).

Usage:

**ated -i ifname # run as a daemon**

### btnd

A daemon program to handle GPIO button event.

Once it detects a button event (click, hold), it call corresponding shell scripts defined under /etc/btnd/ .

Usage:

**btnd <button-name> <gpio-number> & # run as a daemon**

Assume that your product has 2 gpio buttons, one for “WiFi WPS”, uses gpio 2, the other for “restore factory settings”, uses gpio 1. Then you can:

btnd wps 2 & // “wps” is a given name chosen by you.

btnd reset 1 & // just make sure the given name matches the script name under /etc/btnd/.

When the user clicked wps button (gpio 2), btnd would call “sh /etc/btnd/wps\_click.sh”.

When the user held reset button (gpio 1), btnd would call “sh /etc/btnd/reset\_hold.sh”.

You can put your own task in <btn-name>\_hold.sh or <btn-name>\_click.sh.

### eth\_mac

Change the Ethernet MAC address in EEPROM.

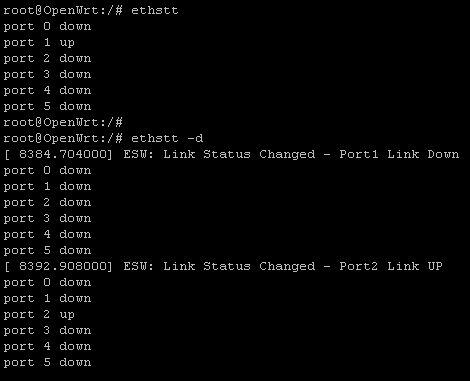
### ethstt

A user tool to query switch port status.

Usage:

**ethstt # print switch port status**

**ethstat –d # run as a daemon**



### gpio

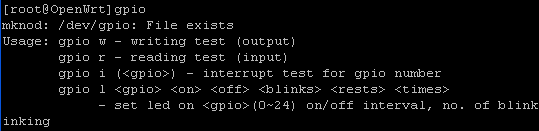
A user tool to query switch port status.

Usage:

**gpio w - writing test (output)**

**gpio r - reading test (input)**

**gpio i (<gpio>) - interrupt test for gpio number**

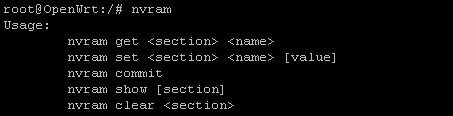
**gpio l <gpio> <on> <off> <blinks> <rests> <times>** 

### nvram

A user tool to manage nvram data.

Usage:

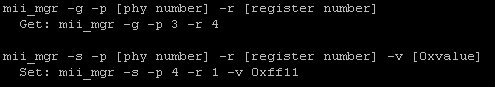
Please check the command help message.



### mii\_mgr

mii register read/wirte test program.

Usage:

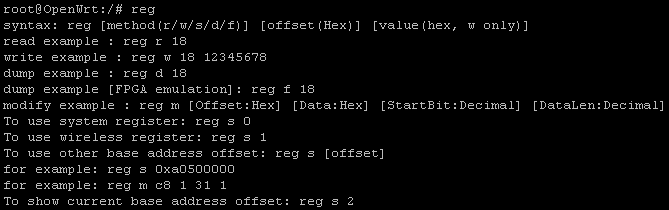


### reg

A user tool to debug system register

Usage:

Please check the command help message.

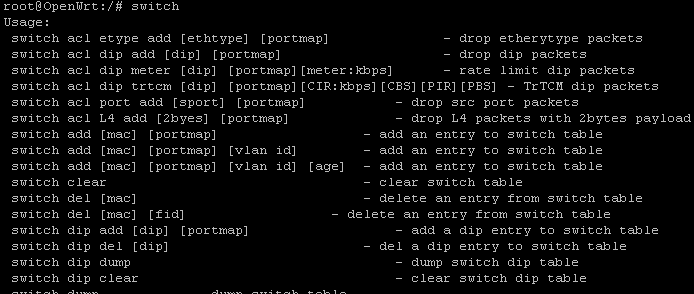


### switch

A user tool to configure Ethernet switch

Usage:

Check the command help message.



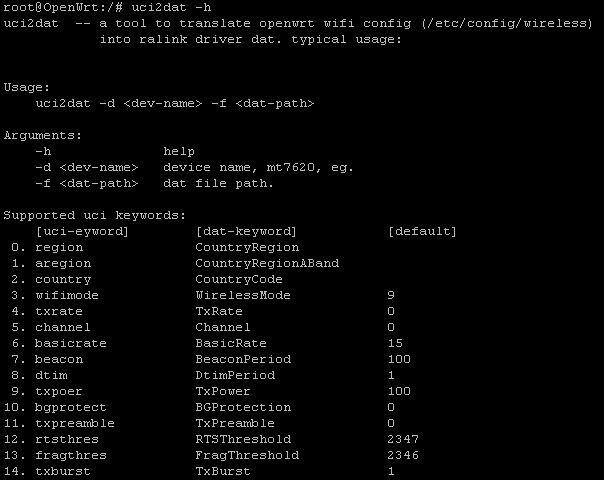
### uci2dat

A user tool to translate OpenWrt uci configuration files into DAT files which can be loaded by Ralink wireless drivers.

Usage:

**uci2dat -h # print help info**

**uci2dat -d devname -f dat filepath # devname is the “device“ name in uci file.**



### watchdog

An user space daemon to co-work with ralink watchdog hardware.

When this program started, it will wake up the watchdog hardware, and it feed the hardware every second. If watchdog program failed to function under some situations (like kernel oops, panic) , the system will be reset by watchdog hardware.

Usage:

**watchdog & # silent daemon**

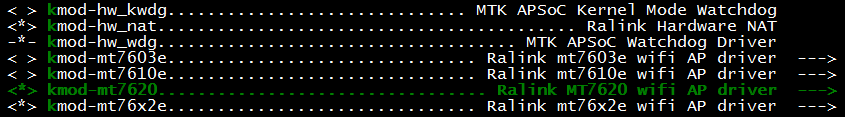
## Drivers

All drivers are put under **{SDKRoot}/dl/** folder.

You can choose the right driver for you device by:

**“make menuconfig” -> “ralink properties” -> “drivers”**

Each driver has its own configuration menu, make sure you didn’t miss the features you want.



These drivers are all built as an OpenWrt packages, you can find the package definition under:

**{SDKRoot}/package/ralink/drivers/mtxxxx/**

├── config.in # the configuration script, can be seen in menuconfig

├── files

│   ├── mt76xx.dat # default driver profile

│   └── mt76xx.sh # startup script, which co-work with OpenWrt framework

├── Makefile # OpenWrt package makefile

└── patches # patches which will be taken in during compiling

├── 001-xxxxxxx.patch

…….

├── xxx-xxxxxxx.patch

### MT7620

WiFi Driver for MT7620.

Driver configuration can be found at package/ramips/driver/mt7620/config.in

### MT7610e

Driver for MT7610e

Driver configuration can be found at package/ramips/driver/mt7610e/config.in

### MT7628/MT7688

Driver for MT7628 and MT7688.

Driver configuration can be found at package/ramips/driver/mt7628/config.in

### MT7603e

Driver for MT7603e

Driver configuration can be found at package/ramips/driver/mt7603e/config.in

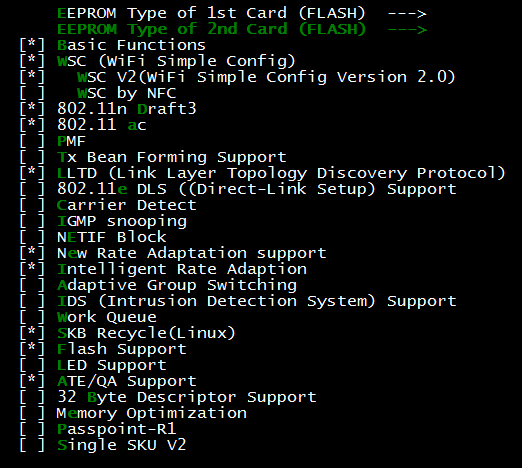
### MT76x2e

All in one driver for MT7602e (2.4G) and MT7612e (5G).

Driver configuration can be found at package/ramips/driver/mt76x2e / config.in

Note: If your MT7612e is onboard, you shoud config EEPROM Type of 2nd Card reading MT7612e’s caldata from flash, if not select efuse.





# Patches

|  |  |  |
| --- | --- | --- |
| Target | Patch Name | Description |
| mt7603e | 001-softirq-warning.patch | Fix a kernel softirq warning |
| mt7620a | 001-fix-kernel-warning.patch | Fix a kernel warning |
| mt7620a | 002-fix-insmod-fail.patch | Insmod fail with 3.10.x kernel |
| mt7620a | 003-support-hwnat.patch | Support hwnat |
| kernel | 0201-firmware-size.patch | firmware size based on flash size |
| kernel | 0301-ramips-profile.patch | 7621 irq profile |
| kernel | 0302-rt-timer.patch | Rt\_timer compile flag |
| kernel | 0305-sdk4.3.0.5\_20141205\_  MT7621\_HW\_NAT.patch | Sync 4300 lsdk 7621 HNAT patch |
| kernel | 0306-sdk4.3.0.10\_20150209\_  6RD\_issue.patch | Sync 4300 lsdk HNAT 6rd issue patch |
| kernel | 0307-to-115775-nand-error.patch | Nand flash squashfs error pacth |
| mt76x2e | 001-led.pacth | 76x2e led support |