

Template:Networking device firmware compile

Introduction

This article contains instructions on how to utilize Teltonika SDK packages for {{{series}}} series devices. The resources used in the provided examples are:

- A {{{device_name}}} {{{device_type}}}
- [Ubuntu 16.04 OS](#)
- SDK version {{{sdk_version}}}

However, the examples apply to other routers, SDK versions and operating systems. In most cases the difference will only be that of a visual nature.

Most examples provided in this page are independent from each other. But it is highly recommended to acquaint yourself with the basics by reading the "[Prerequisites](#)" and "[Compiling a standard firmware](#)" sections first as they contain information that will be necessary in order to understand some aspects of the other examples.

It should also be noted early that the first time you compile a firmware it may take **about two hours** to complete. On later attempts the duration is considerably lower.

Prerequisites

You will need:

- A PC, laptop or virtual machine running Linux OS (We recommend using Ubuntu 16.04 LTS)
- A {{{series}}} series {{{device_type}}}
- An SDK intended for your router, which can be downloaded here: [Software Development Kit](#)

Compiling a standard firmware

First, you must install the packages required for the SDK to work. Open the Terminal application (**Ctrl + ALT + T**) and execute the following commands:

```
sudo apt update
```

```
sudo apt install -y gcc binutils bzip2 flex python3.6 perl make grep  
diffutils unzip gawk subversion zlib1g-dev build-essential u-boot-tools
```

Create a new folder (for this example I'll be using a directory called {{{dev_dir}}} found at my **hard drive** directory) and extract the SDK archive inside it. You can achieve this with drag and drop or by executing this command via Terminal:

```
tar -xf ~/Downloads/{{{sdk_device_name}}}_R_GPL_00.XX.YY.Z.tar.gz -C  
~/{{{dev_dir}}}
```

Note: don't forget to replace the file name and path in accordance with your own circumstances

Open a Terminal inside the SDK directory. You can change the directory in your current terminal (*cd ~/{{{dev_dir}}}/openwrt-gpl-{{{arch_type}}}.Linux-x86_64*).

Once you open the Terminal you can compile a standard firmware by executing this command in the terminal:

```
make
```

If all is in order, the output should look something like this:

```
[[File:{{{compile_no_threads_file}}}|border|class=tlr-border|1000 px]]
```

Note: the first time you compile a firmware file it may take up to **two hours** before it is complete. Don't close the Terminal window up until then. Once it is finished, you will find the firmware in the *./bin/targets/{{{arch_type}}}/generic/tlrFws* directory. It should contain a file *{{{sdk_device_name}}}_R_GPL_00.XX.YY.Z_WEBUI.bin*, it can be used to upgrade your router's firmware via its web interface.

You can speed up this process by passing an extra argument **-j**. This argument then compiles the necessary packages in parallel. To know how much jobs you can pass to the compiling process execute this command in the terminal:

```
nproc
```

This commands output should be a number. This number tells how much processing units are available to the current process. Now execute this command in the terminal:

```
make -j<nproc_output>
```

```
[[File:{{{compile_with_threads_file}}}|border|class=tlr-border|1000 px]]
```