

WF111 LINUX DRIVER INSTALLATION

APPLICATION NOTE

Thursday, 24 May 2012

Version 1.0



Copyright © 2001 - 2012 Bluegiga Technologies

Bluegiga Technologies reserves the right to alter the hardware, software, and/or specifications detailed herein at any time without notice, and does not make any commitment to update the information contained herein. Bluegiga Technologies assumes no responsibility for any errors which may appear in this manual. Bluegiga Technologies' products are not authorized for use as critical components in life support devices or systems.

Bluegiga Access Server, Access Point, AX4, BSM, iWRAP, BGScript and WRAP THOR are trademarks of Bluegiga Technologies.

The *Bluetooth* trademark and logo are registered trademarks and are owned by the Bluetooth SIG, Inc.

ARM and ARM9 are trademarks of ARM Ltd.

Linux is a trademark of Linus Torvalds.

All other trademarks listed herein belong to their respective owners.

TABLE OF CONTENTS

1. Introduction	4
2. Prerequisites	5
2.1 Prerequisites	5
2.1.1 Needed files	5
3. Compiling and installing WF111 driver	6
3.0.2 Building kernel modules for ARM	6
3.0.3 Building kernel modules for x86/i386	6
3.0.4 Overriding the MAC address	7
3.0.5 Installing the driver files	7
4. Verifying driver installation	8
5. Troubleshooting	9
5.0.6 Troubleshooting	9
6. Contact information	12

1 Introduction

This document describes how to compile and install the Linux drivers for Bluegiga WF111 802.11 b/g/n module and how to verify the driver functionality.

2 Prerequisites

2.1 Prerequisites

Linux MMC/SD/SDIO driver support requires kernel version 2.6.24 up to v2.6.37. Your systems kernel version can be checked by:

```
uname -a

Linux ubuntu 2.6.37-13-generic Ubuntu SMP Wed Nov 2 13:27:26 UTC
2011 i386 GNU/Linux
```

For client mode (STA) with WPA/WPA2 support requires wireless-tools version 28 or newer and wpa_supplicant. This can be checked using the command "iwconfig -v".



In the Linux kernel configuration you must have the following options enabled:

- CONFIG_WIRELESS_EXT
- CONFIG_MODULES
- CONFIG_FW_LOADER

On Ubuntu this can be viewed with the command `less /boot/config-$(uname -r)`

Userspace must support firmware loading, in particular when using an embedded system with Busybox and its mdev, the option `ENABLE_FEATURE_MDEV_LOAD_FIRMWARE` must be set. On a non-busybox system this is normally handled by udev.

2.1.1 Needed files


- WF111 Linux kernel module source: **csr-linux-wifi-5.0.3-oss.tar.gz**
 - Source code for `unifi_sdio.ko`
 - Minor patch for `wpa_supplicant` to allow seamless roaming (i.e. roaming between APs in the same ESS with the same SSID) and PMK caching to work correctly. The patch provides Information Elements relating to the new AP following a roam. The patch for v0.6.8 can also be applied to later v0.6.x `wpa_supplicant` releases. The patch command will report an error patching the version header file, but the patch will otherwise succeed.
 - Source code for optional CSR optimized sdio drivers.
- The WF111 userspace applications. For ARM: **unifi-helper_5.0.3-r0_armv5te.tar.gz**. For x86/i386: **unifi-helper_5.0.3-r0_x86.tar.gz**
 - **unififw** - The script that runs `unifi_helper` with the appropriate parameters when a WF111 module is detected. This gets automatically called by the kernel module to start the `unifi_helper`.
 - **unifi_helper** - User-space helper daemon for the WF111 driver, started by the `unififw`.
 - **unifi_config** - Configuration and status reporting utility, used for example to configure power saving modes.
- The firmware for the WF111 module: **firmware_5.0.3-r0_all.tar.gz**
 - **sta.xbv** - Firmware executed in the Wi-Fi module
 - **ufmib.dat** - Configuration parameters used by the Firmware such as power tables.
 - **mac.txt** - A way to override the Wi-Fi MAC address. This is not needed as the MAC is stored inside the WF111 module.

3 Compiling and installing WF111 driver


3.0.2 Building kernel modules for ARM

In the directory where you have copied the required files, issue the following commands to compile the kernel module:

```
mkdir -p unifi-gpl/output
cd unifi-gpl
tar xvf ../csr-linux-wifi-5.0.3-oss.tar.gz
tar xvf ../unifi-helper_5.0.3-r0_armv5te.tar.gz -C output
tar xvf ../firmware_5.0.3-r0_all.tar.gz -C output
cd csr_wifi_5.0.3
fakeroot ./linux_driver_install mmc wext_ap install ARCH=arm
DESTDIR=$(pwd)/../output KDIR=path/to/arm_kernel/source
CROSS_COMPILE=/usr/bin/arm-linux-
```

 The **DESTDIR** is the location where the kernel module will be placed. The **CROSS_COMPILE** is the prefix for the arm tools, so for example if the arm gcc is `"/usr/bin/arm-linux-gcc"`, then the prefix would be `"/usr/bin/arm-linux-"`.


The **KDIR** is the path to where the rest of the Linux kernel sources are. You need to change the `"path/to/arm_kernel/source"` to match the location of ARM Linux kernel sources.

 The script will claim to fail "FATAL: Could not open /lib/modules for writing". This is because it attempts to install them locally which you don't want to do when crosscompiling. The files are also placed in the `"../output"` directory from where you can copy them to your target system.

3.0.3 Building kernel modules for x86/i386

In the directory where you have copied the required files, issue the following commands to compile the kernel module:

```
mkdir -p unifi-gpl/output
cd unifi-gpl
tar xvf ../csr-linux-wifi-5.0.3-oss.tar.gz
tar xvf ../unifi-helper_5.0.3-r0_x86.tar.gz -C output
tar xvf ../firmware_5.0.3-r0_all.tar.gz -C output
cd csr_wifi_5.0.3
fakeroot ./linux_driver_install mmc wext_ap install ARCH=i386
DESTDIR=$(pwd)/../output KDIR=/usr/src/linux-headers-$(uname -r)
```

 The **DESTDIR** is the location where the kernel module will be placed.

The **KDIR** is the path to where the rest of the Linux kernel sources are.



The script will claim to fail "FATAL: Could not open /lib/modules for writing". This is because it attempts to install them directly which fakeroot will not allow. The files are placed in the "../output" directory from where you can copy them to your target system.

3.0.4 Overriding the MAC address

To override the MAC address you need to write a file called */lib/firmware/unifi-sdio-0/mac.txt*, which will be located in the *DESTDIR* directory after the previous commands. If there is no need to override the MAC address, *mac.txt* should be deleted.

3.0.5 Installing the driver files

Copy the full file system hierarchy in the output folder to target machine. The files in the output directory should be:

- /lib/modules/KERNEL_VERSION/extra/unifi_sdio.ko
- /lib/firmware/unifi-sdio-0/ufmib.dat
- /lib/firmware/unifi-sdio-0/sta.xbv
- /usr/sbin/unififw
- /usr/sbin/unifi_helper
- /usr/sbin/unifi_config

By default the firmware files will be in */lib/firmware/unifi-sdio-0/* which assumes the WF111 is connected to the host's first SDIO slot. However if the WF111 is connected to the second slot, the firmware files need to be placed in */lib/firmware/unifi-sdio-1/*. Upon startup the driver will print out lines beginning with "*unifi0:*" if it is in the first slot and "*unifi1:*" if it is in the second slot.

On the target system, execute the command below to load the WF111 driver into the kernel.

```
depmod -a
modprobe unifi_sdio
```

4 Verifying driver installation

The WF111 driver will print some debug messages when loaded, ending with the "UniFi ready" message. These messages can be seen using the `dmesg` command, or by viewing the syslog, normally located at `/var/log/syslog` or `/var/log/messages`.

```
unifi0: UniFi f/w protocol version 9.1 (driver 9.1)
unifi0: Firmware build 1089: 2010-10-05 14:50
cindr03_core_softmac_rom_sdio_gcc 1089 bfw@eagle@630492
unifi0: Firmware patch 1247
unifi0: unifi0 is eth1
unifi0: UniFi ready
```

Please note that unless changed, the WF111 kernel module will call the device **ethX** where X is the interface number, and not **wlanX**.

You can test that Wi-Fi access points are found using the **iwlist** command.

```
iwlist scan
```

For more information about Linux wireless tools, please refer for example to:

- http://en.wikipedia.org/wiki/Wireless_tools_for_Linux
- http://linux.die.net/man/5/wpa_supplicant.conf

5 Troubleshooting

Before starting to troubleshooting, please verify that the prerequisites are full-filled especially the Linux kernel configuration as described in the Prerequisites section.

5.0.6 Troubleshooting

MAC fault

On occasion when the driver is loaded, a MAC fault message will be generated. The driver will automatically recover from this.

```
unifi0:      543633: MAC fault 0030, arg 0924 (x6)
```

Block write failed

If the EEPROM is faulty, the module's firmware may fail to start. The symptom of this is a quick stream of error messages.

```
UniFi SDIO Driver: 5.0.1 Sep  2 2011 13:25:15 CSR SME with WEXT support Kernel 3.1.0
```

```
UniFi: Using native Linux MMC driver for SDIO.
```

```
sdio bus_id:      mmc0:0001:1 - UniFi card 0x0 inserted
```

```
unifi0: Initialising UniFi, attempt 1
```

```
unifi0: Chip ID 0x07  Function 1  Block Size 512  Name UniFi-4(UF60xx)
```

```
unifi0: Chip Version 0x3A22
```

```
unifi0: Calling CsrSdioHardReset
```

```
unifi0: Falling back to software hard reset
```

```
unifi0: Chip ID 0x07  Function 1  Block Size 512  Name UniFi-4(UF60xx)
```

```
unifi0: MAILBOX2 non-zero after reset (mbox2 = ffff)
```

```
unifi0: unifi_dl_patch c2c4de98 0100060e
```

```
unifi0: Block write failed
```

```
unifi0: CMD53 failed writing 2042 bytes to handle 1
```

```
unifi0: Failed to copy block of 2042 bytes to UniFi
```

```
unifi0: Patch failed after 0 bytes
```

```
unifi0: Failed to patch image
```

```
unifi0: Failed to patch firmware
```

```
unifi0: Failed to establish communication with UniFi
```

```
unifi0: Failed to start host protocol.
```

```
unifi0: Failed to initialise UniFi chip.
```

```
unifi0: Initialising UniFi, attempt 2
```

```
unifi0: Chip ID 0x07  Function 1  Block Size 64  Name UniFi-4(UF60xx)
```

```
unifi0: Chip Version 0x3A22
```

```
unifi0: Calling CsrSdioHardReset
```

```
unifi0: Falling back to software hard reset
```

```
unifi0: Chip ID 0x07  Function 1  Block Size 64  Name UniFi-4(UF60xx)
```

```
unifi0: MAILBOX2 non-zero after reset (mbox2 = ffff)
```

```
unifi0: unifi_dl_patch c2c4de98 0100060e
```

```
unifi0: Block write failed
```

```
unifi0: CMD53 failed writing 2042 bytes to handle 1
```

```
unifi0: Failed to copy block of 2042 bytes to UniFi
```

```
unifi0: Patch failed after 0 bytes
```

```
unifi0: Failed to patch image
```

```
unifi0: Failed to patch firmware
```

No "unifi0: ..." messages appear in system log file

When the driver is loaded using modprobe the following is output to the system log:

```
UniFi SDIO Driver: 5.0.1 Sep  2 2011 13:25:15
```

```
CSR SME with WEXT support
```

```
Kernel 3.1.0
```

```
UniFi: Using native Linux MMC driver for SDIO.
```

However nothing else gets printed, i.e. the unifi0 driver doesn't detect the card.

When the WF111 evaluation kit is not detected in the SDIO, this typically means that it is not getting power. Please check the jumper configuration.

Debugging for more info

By using the module with a higher debug level, much more information can be obtained.

To enable more verbose debugging, issue commands:

```
rmmmod unifi_sdio
modprobe unifi_sdio unifi_debug=3
```

An example trace of WF111 initialization log is below.

```
unifil: UDI 0 (0xc3b55fa8) registered. configuration = 0x0
unifil: Netdev c3b54000 client (id:0 s:0xC000) is registered
unifi: uf_netdev_event: ignore e=5, ptr=c3807c00, priv=c3807f40 lo
unifi: uf_netdev_event: ignore e=1, ptr=c3807c00, priv=c3807f40 lo
unifi: uf_netdev_event: ignore e=5, ptr=c3920800, priv=c3920b40 eth0
unifi: uf_netdev_event: ignore e=1, ptr=c3920800, priv=c3920b40 eth0
unifil: Allocate buffers for 5 core dumps
unifil: Core dump configured (5 dumps max)
unifil: CsrWifiRouterTransportInit:
unifil: run UniFi helper app...
unifil: starting /usr/sbin/unififw
unifil: running /usr/sbin/unififw 1 2
unifil: UDI 1 (0xc3b56004) registered. configuration = 0x22
unifil: SME client (id:1 s:0xC100) is registered
unifil: UNIFI_BUILD_TYPE userspace=AP
unifil: CsrWifiRouterCtrlWifiOnReqHandler(0x0003)
SDIO: Skip power on; card is already powered.
unifil: Initialising UniFi, attempt 1
unifil: Resetting UniFi
unifil: Chip ID 0x07&nbsp; Function 1&nbsp; Block Size 512&nbsp; Name
UniFi-4(UF60xx)
unifil: Block mode SDIO
unifil: Chip Version 0x3A22
unifil: Calling CsrSdioHardReset
unifil: Falling back to software hard reset
unifil: Chip ID 0x07&nbsp; Function 1&nbsp; Block Size 512&nbsp; Name
UniFi-4(UF60xx)
unifil: Hard reset (IO_ENABLE)
unifil: waiting for disable to complete, attempt 0
unifil: Disable complete (function 1 is disabled) in ~ 0 msecs
unifil: waiting for reset to complete, attempt 0
unifil: Reset complete (function 1 is disabled) in ~ 0 msecs
unifi: Set SDIO function block size to 64
unifil: unifi_configure_low_power_mode: new mode = disabled, wake_host = FALSE
unifil: unifi_run_bh: discard message.
unifil: waiting for MAILBOX1 to be non-zero...
unifil: MAILBOX1 ready (0x0100) in 0 millisecs
unifil: MAILBOX1 value=0x0100
...
```

6 Contact information

Sales: sales@bluegiga.com

Technical support: support@bluegiga.com
<http://techforum.bluegiga.com>

Orders: orders@bluegiga.com

WWW: <http://www.bluegiga.com>
<http://www.bluegiga.hk>

Head Office / Finland: Phone: +358-9-4355 060
Fax: +358-9-4355 0660
Sinikalliontie 5 A
02630 ESPOO
FINLAND

Head address / Finland: P.O. Box 120
02631 ESPOO
FINLAND

Sales Office / USA: Phone: +1 770 291 2181
Fax: +1 770 291 2183
Bluegiga Technologies, Inc.
3235 Satellite Boulevard, Building 400, Suite 300
Duluth, GA, 30096, USA

Sales Office / Hong-Kong: Phone: +852 3182 7321
Fax: +852 3972 5777
Bluegiga Technologies, Inc.
19/F Silver Fortune Plaza, 1 Wellington Street,
Central Hong Kong