



ABSTRACT

This document is intended for collecting additional logs for debugging purposes. This is in continuation to the debug FAQs presented in the *FAQ* and *Debug Hints* section of the [WiLink8 Linux Wi-Fi Driver Release R8.8 Build User's Guide](#). The following logs will help to debug issues that may arise during product development and Wi-Fi® driver integration.

Table of Contents

1 Acquire WL18xx Firmware Version	2
2 Enable wlcore Driver Kernel Message	2
2.1 Debug Parameter Options.....	2
2.2 Activation of Debug Printout.....	2
2.3 Additional Examples to Enable Debug Message Groups.....	3
3 Enable Dynamic Debug for wl18xx, mac80211, cfg80211	3
3.1 Checking for the Dynamic Debug Compile Switch	3
3.2 Enabling Dynamic Debug	4
4 Enabling and Disabling WiLink8 Features	5
4.1 DFS Channel Related Commands	5
4.2 Enhanced Low Power Enable/Disable Commands	5

Trademarks

Wi-Fi® is a registered trademark of Wi-Fi Alliance.
All trademarks are the property of their respective owners.

1 Acquire WL18xx Firmware Version

In many cases it helps to know the version of the firmware being used to provide the needed input to debug an issue. The below command prints the firmware version

```
root@am335x-evm:~# ifconfig wlan0 down
root@am335x-evm:/usr/share/wl18xx# ifconfig wlan0 up
[ 2077.084115] wlcore: PHY firmware version: Rev 8.2.0.0.245
[ 2077.180632] wlcore: firmware booted (Rev 8.9.0.0.85)
```

2 Enable wlcore Driver Kernel Message

wlcore module provides options to enable additional debugging from the default prints that are available as part of the status to a specific Wi-Fi operation (staring a STA, AP and so forth or disabling the same). It may however be needed to have additional debug logs while debugging certain operation of the driver. These logs can be dynamically enabled while running the driver. The following section specifies the available dynamic debug groups and the method to enable them. The example section shows how a collection of these dynamic debugs can be enabled using a single command.

2.1 Debug Parameter Options

The WL18xx driver supports configurable and very detailed debug information. The log system of WL18xx driver only prints that log output that has an appropriate debug mode. All other debug logs are not printed. In this way, it is possible to control only necessary debug information. All available debug modes are defined in the **drivers/net/wireless/ti/wlcore/debug.h** file. The options are listed below for reference. It is recommended to refer to the above mentioned header file for accurate debug groups available for a specific version of the driver used in the build.

```
DEBUG_NONE           = 0,
DEBUG_IRQ            = BIT(0), /* 1 */
DEBUG_SPI            = BIT(1), /* 2 */
DEBUG_BOOT           = BIT(2), /* 4 */
DEBUG_MAILBOX        = BIT(3), /* 8 */
DEBUG_TESTMODE       = BIT(4), /* 16 */
DEBUG_EVENT          = BIT(5), /* 32 */
DEBUG_TX             = BIT(6), /* 64 */
DEBUG_RX             = BIT(7), /* 128 */
DEBUG_SCAN           = BIT(8), /* 256 */
DEBUG_CRYPT          = BIT(9), /* 512 */
DEBUG_PSM            = BIT(10), /* 1024 */
DEBUG_MAC80211       = BIT(11), /* 2048 */
DEBUG_CMD            = BIT(12), /* 4096 */
DEBUG_ACX            = BIT(13), /* 8192 */
DEBUG_SDIO           = BIT(14), /* 16384 */
DEBUG_FILTERS        = BIT(15), /* 32768 */
DEBUG_ADHOC          = BIT(16), /* 65536 */
DEBUG_AP             = BIT(17), /* 131072 */
DEBUG_PROBE          = BIT(18), /* 262144 */
DEBUG_IO             = BIT(19), /* 524288 */
DEBUG_MASTER         = (DEBUG_ADHOC | DEBUG_AP), /* 196608 */
DEBUG_ALL            = ~0,
```

2.2 Activation of Debug Printout

The following sections detail the method to enable dynamic debug message groups within the driver. The following command is used for instructing the kernel to send log messages to the console:

```
echo 8 > /proc/sys/kernel/printk
```

To configure debug mode of the driver, write one of the above debug category number to `/sys/module/wl12xx/parameters/debug_level` file. For example, following command will configures driver to print debug messages related to scanning operation - `DEBUG_SCAN`.

```
echo 256 > /sys/module/wlcore/parameters/debug_level
```

Alternatively, you may enter hex number instead of decimal.

```
echo 0x100 > /sys/module/wlcore/parameters/debug_level
```

Table 2-1 shows the output prints when the scan command is executed after starting the STA.

Table 2-1. Typical Debug Output During Scan

```
root@am335x-evm:~# iw wlan0 scan > /dev/null
[ 3132.079650] wlcore: normal scan role_id 0
[ 3132.083862] wlcore: freq 2467, ch. 012, flags 0x36, power 20, min/max_dwell 25/50, PASSIVE
[ 3132.092529] wlcore: freq 2472, ch. 013, flags 0x36, power 20, min/max_dwell 25/50, PASSIVE
[ 3132.101165] wlcore: freq 2484, ch. 014, flags 0x36, power 20, min/max_dwell 25/50, PASSIVE
[ 3132.109802] wlcore: freq 2412, ch. 001, flags 0x20, power 20, min/max_dwell 25/50
[ 3132.117614] wlcore: freq 2417, ch. 002, flags 0x20, power 20, min/max_dwell 25/50
[ 3132.125427] wlcore: freq 2422, ch. 003, flags 0x20, power 20, min/max_dwell 25/50
[ 3132.133239] wlcore: freq 2427, ch. 004, flags 0x20, power 20, min/max_dwell 25/50
[ 3132.141052] wlcore: freq 2432, ch. 005, flags 0x00, power 20, min/max_dwell 25/50
[ 3132.148864] wlcore: freq 2437, ch. 006, flags 0x00, power 20, min/max_dwell 25/50
[ 3132.156677] wlcore: freq 2442, ch. 007, flags 0x00, power 20, min/max_dwell 25/50
[ 3132.164489] wlcore: freq 2447, ch. 008, flags 0x10, power 20, min/max_dwell 25/50
[ 3132.172332] wlcore: freq 2452, ch. 009, flags 0x10, power 20, min/max_dwell 25/50
[ 3132.180145] wlcore: freq 2457, ch. 010, flags 0x10, power 20, min/max_dwell 25/50
[ 3132.187957] wlcore: freq 2462, ch. 011, flags 0x10, power 20, min/max_dwell 25/50
[ 3132.195770] wlcore: 2.4GHz: active 11 passive 3
[ 3132.200866] wlcore: 5GHz: active 0 passive 0
[ 3132.205688] wlcore: DFS: 0
[ 3132.208862] wlcore: build probe request band 0.
```

2.3 Additional Examples to Enable Debug Message Groups

A group of messages can also be enabled depending on the issue being debugged. The examples below show that you can enable a group of messages. The required flags can be ORed together to create the value to be written into the debug_level.

Enable debug printout during Connection

```
echo 0x63c00 > /sys/module/wl12xx/parameters/debug_level
```

This will enable the following debug level (as defined in the drivers/net/wireless/wl18xx/wl18xx.h file):

```
DEBUG_PSM      = BIT(10),
DEBUG_MAC80211 = BIT(11),
DEBUG_CMD      = BIT(12),
DEBUG_ACX      = BIT(13),
DEBUG_AP       = BIT(17),
DEBUG_MASTER   = (DEBUG_ADHOC | DEBUG_AP),
```

3 Enable Dynamic Debug for wl18xx, mac80211, cfg80211

The WL18xx driver uses the Linux kernel dynamic debug mechanism (debugfs), but it implements its own interface above. To enable debug output, the whole module has to be enabled in the dynamic debug Linux kernel subsystem and configuring the specific debug information one might be interested in is done via driver debug categories.

3.1 Checking for the Dynamic Debug Compile Switch

Ensure that the kernel is built with the dynamic debug flags enabled. The compilation switch “CONFIG_DYNAMIC_DEBUG” should be set to “Y” before enabling the dynamic debug options.

```
root@am335x-evm:~# cat /proc/config.gz | gunzip | grep CONFIG_DYNAMIC_DEBUG
# CONFIG_DYNAMIC_DEBUG=y
```

3.2 Enabling Dynamic Debug

To configure debug output using dynamic debug subsystem, the following steps have to done:

1. Mount debugfs (this should have been done by system automatically):
2. Enable kernel print out
3. Enable debug output for either of wl18xx, mac80211, cfg80211 modules, or for all of them.
4. Write the required debug category to /sys/module/wl12xx/parameters/debug_level file.

Enable the dynamic debug statistics type the following after bring the wlan0 interface up.

Table 3-1. Commands to Enable Dynamic Debug

```

root@am335x-evm:~# ifconfig wlan0 up
[382.611846] wlcore: PHY firmware version: Rev 8.2.0.0.212 [382.708557] wlcore: firmware booted (Rev 8.9.0.0.17)
root@am335x-evm:~# mount -t debugfs none /sys/kernel/debug root@am335x-evm:~# echo 8 > /proc/sys/kernel/printk
root@am335x-evm:~# echo 0x8800 > /sys/module/wlcore/parameters/debug_level root@am335x-evm:~# echo -n 'module wlcore +p'
> /sys/kernel/debug/dynamic_debug/control
root@am335x-evm:~# echo -n 'module wl18xx +p' > /sys/kernel/debug/dynamic_debug/control
root@am335x-evm:~# echo -n 'module mac80211 +p' > /sys/kernel/debug/dynamic_debug/control
root@am335x-evm:~# echo -n 'module cfg80211 +p' > /sys/kernel/debug/dynamic_debug/control
  
```

Navigate to interface debug directory. You can see debug information by performing "cat" on any files. In this example, we performed "cat driver_state" to exam driver information.

Table 3-2. Check for Enabled Messages Written to File

```

root@am335x-evm:~# cd /sys/kernel/debug/ieee80211/phy0/wlcore/
root@am335x-evm:/sys/kernel/debug/ieee80211/phy0/wlcore# ls
beacon_filtering gpio_power split_scan_timeout beacon_interval irq_blk_threshold start_recovery driver_state irq_pkt_threshold
stats_tx_aggr dtim_interval irq_timeout suspend_dtim_interval dynamic_ps_timeout mem tx_queue_len excessive_retries retry_count
vifs_state forced_ps rx_streaming wl18xx_fw_stats_raw sleep_auth
root@am335x-evm:/sys/kernel/debug/ieee80211/phy0/wlcore# cat driver_state tx_blocks_available = 44
tx_allocated_blocks = 0 .
..
chip.fw_ver_str = Rev 8.9.0.0.85
chip.phy_fw_ver_str = Rev 8.2.0.0.245
recovery_count = 0
  
```

Similarly the firmware debug directory contains firmware debug information.

```

root@am335x-evm:/sys/kernel/debug/ieee80211/phy0/wlcore# cd wl18xx/fw_stats/
  
```

4 Enabling and Disabling WiLink8 Features

In some cases it may be handy to enable or disable certain features to check for the interdependency of these features with an issue being debugged. The following sections specifies the ways to enable or disable certain features in the WiLink8.

4.1 DFS Channel Related Commands

Reset DFS channels state:

```
echo 1 > /sys/kernel/debug/ieee80211/phy0/dfs_reset
```

The number is a bitmap:

bit0 - reset all AVAILABLE dfs channels (to USABLE)

bit1 - reset all UNAVAILABLE dfs channels (to USABLE)

Trigger a debug radar detection command on a channel:

```
echo <num> > /sys/kernel/debug/ieee80211/phy0/wlcore/wl18xx/radar_detection
```

<num> is the DFS channel number. Ex: 60.

Note:

- Trying to trigger radar on non-operational channel will results with no command.
- For 40 Mhz: trigger the radar on the primary channel only.

4.2 Enhanced Low Power Enable/Disable Commands

Disable ELP	<code>iw wlan0 set power_save off echo 0 > /sys/kernel/debug/ieee80211/phy0/wlcore/sleep_auth</code>
Enable ELP	<code>iw wlan0 set power_save on echo 2 > /sys/kernel/debug/ieee80211/phy0/wlcore/sleep_auth</code>

IMPORTANT NOTICE AND DISCLAIMER

TI PROVIDES TECHNICAL AND RELIABILITY DATA (INCLUDING DATA SHEETS), DESIGN RESOURCES (INCLUDING REFERENCE DESIGNS), APPLICATION OR OTHER DESIGN ADVICE, WEB TOOLS, SAFETY INFORMATION, AND OTHER RESOURCES "AS IS" AND WITH ALL FAULTS, AND DISCLAIMS ALL WARRANTIES, EXPRESS AND IMPLIED, INCLUDING WITHOUT LIMITATION ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT OF THIRD PARTY INTELLECTUAL PROPERTY RIGHTS.

These resources are intended for skilled developers designing with TI products. You are solely responsible for (1) selecting the appropriate TI products for your application, (2) designing, validating and testing your application, and (3) ensuring your application meets applicable standards, and any other safety, security, regulatory or other requirements.

These resources are subject to change without notice. TI grants you permission to use these resources only for development of an application that uses the TI products described in the resource. Other reproduction and display of these resources is prohibited. No license is granted to any other TI intellectual property right or to any third party intellectual property right. TI disclaims responsibility for, and you will fully indemnify TI and its representatives against, any claims, damages, costs, losses, and liabilities arising out of your use of these resources.

TI's products are provided subject to [TI's Terms of Sale](#) or other applicable terms available either on ti.com or provided in conjunction with such TI products. TI's provision of these resources does not expand or otherwise alter TI's applicable warranties or warranty disclaimers for TI products.

TI objects to and rejects any additional or different terms you may have proposed.

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265
Copyright © 2022, Texas Instruments Incorporated