WB Logging and Troubleshooting

WB45NBT and WB50NBT

Application Note *v1.0*

INTRODUCTION

Laird's WB products (WB40NBT, WB45NBT, and WB50NBT) are full Linux systems and come pre-equipped with a wide array of logging and troubleshooting tools. This application note explains how to work with these tools to help integrate a WB product into your environment.

REQUIREMENTS

The document assumes you're working with a WB45NBT or WB50NBT mounted on a BB45/BB40 development board and running GA5 release software. Due to varying hardware revisions and software revisions, different tools or access methods may be required. This application note attempts to cover the most general procedures.

INTERFACES

Troubleshooting resources vary across the WB's interfaces. The widest set of tools is available from the command console. Depending on your device's configuration, you may access the command console one of two ways:

- Connect over the DEBUG UART serial port interface
- Log in via SSH over any network interface (Wi-Fi, Ethernet, or USB Ethernet Gadget)

Laird's WebLCM browser application provides many useful logging and troubleshooting tools. WebLCM is available on the following interfaces:

- Over HTTPS when connected to the WB over Wi-Fi in AP mode
- Over HTTPS when both the WB and your host computer are connected over Wi-Fi to the same network
- Over HTTPS on the Ethernet interface
- Over HTTPS on the USB Ethernet interface

Note: WebLCM is not available in all versions of WB software.

LOGGING

The WB provides several logging functions for diagnostic purposes. The following logs are active by default:

Americas: +1-800-492-2320
Europe: +44-1628-858-940
Hong Kong: +852 2923 0610

- syslog – syslog is the primary combined log of Linux. Many different daemons and programs report to the main system log. Much of the kernel log output also goes into this log. The supplicant (sdcsupp), event monitor (event_mon), and SDK applications all log to the syslog.
- Kernel log – The kernel log is used by the kernel, drivers, and various system components. The kernel log is accessed by the program dmesg. By default, WARNING (level 4) and higher messages are printed to the log.

These two logs are easily accessible by either the WebLCM (if available) or the console. The steps to access these logs are detailed in the following sections.

WebLCM

To access the log output through WebLCM, follow these steps:

1. Enter the IP address of your WB in a web browser.
2. Enter your login information. By default, the user name is **root** and the password is **summit**.
3. Click the **About** tab (Figure 1).

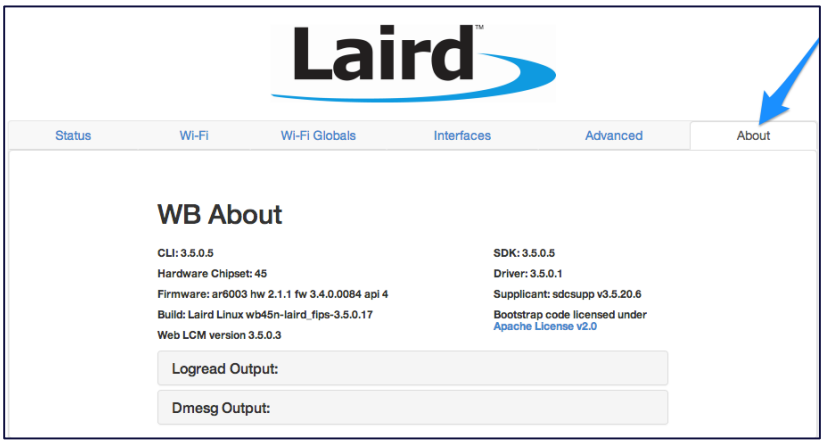

Figure 1: WebLCM About tab

On this page, you can find both logs. The syslog is available in the *Logread Output* rollout and the kernel log is available in the *Dmesg Output* rollout.

4. Click on one of these headers to open the respective log.

Console

Both logs are also available at the command console. From any console (DEBUG UART, SSH login, or similar), enter commands; the log directly outputs to the console.

Syslog

To read the syslog, enter **logread** and press **Return**. The output should resemble the following:

```
Jan  1 00:41:23 summit user.info usbmount[749]: executing command: run-parts
Jan  3 01:22:29 summit user.info    nis: wl down stop
Jan  3 01:22:30 summit user.info    nis: wl up start
Jan  3 01:22:31 summit user.warn kernel: ath6kl: INIT GENERIC NETLINK ATHEROS COM
Jan  3 01:22:31 summit user.info kernel: ath6kl: ar6003 hw 2.1.1 sdio fw 3.4.0.0084^A api 4
Jan  3 01:22:33 summit user.info laird[1464]: Laird Event Monitor Version 1.1.2
Jan  3 01:22:33 summit user.info laird[1464]: Current Registered Bitmask 0x0000003FA3008000
```


Americas: +1-800-492-2320
Europe: +44-1628-858-940
Hong Kong: +852 2923 0610

Kernel Log

To read the kernel log, enter **dmesg** and hit return. The output should resemble the following:

```
usb0: MAC 7e:07:10:8c:e5:e5
usb0: HOST MAC 4a:24:fc:b7:0d:41
 gadget: Ethernet Gadget, version: Memorial Day 2008
 gadget: g ether ready
ath6kl: INIT GENERIC NETLINK ATHEROS COM
ath6kl: ar6003 hw 2.1.1 sdio fw 3.4.0.0084\x01 api 4
nf conntrack version 0.5.0 (947 buckets, 3788 max)
macb f802c000.ethernet eth0: link up (100/Full)
ip tables: (C) 2000-2006 Netfilter Core Team
cfg80211: Calling CRDA for country: US
cfg80211: Regulatory domain changed to country: US
cfg80211:   (start freq - end freq @ bandwidth), (max antenna gain, max eirp)
cfg80211:   (2402000 KHz - 2472000 KHz @ 40000 KHz), (300 mBi, 2700 mBm)
cfg80211:   (5170000 KHz - 5250000 KHz @ 40000 KHz), (300 mBi, 1700 mBm)
cfg80211:   (5250000 KHz - 5330000 KHz @ 40000 KHz), (300 mBi, 2000 mBm)
cfg80211:   (5490000 KHz - 5600000 KHz @ 40000 KHz), (300 mBi, 2000 mBm)
cfg80211:   (5650000 KHz - 5710000 KHz @ 40000 KHz), (300 mBi, 2000 mBm)
cfg80211:   (5735000 KHz - 5835000 KHz @ 40000 KHz), (300 mBi, 3000 mBm)
cfg80211:   (57240000 KHz - 63720000 KHz @ 2160000 KHz), (N/A, 4000 mBm)
```

log_dump

A utility to collect both logs and other important information about the system is provided in the form of **/usr/bin/log_dump**.

When run, a lug_dump.txt file is created along with a message about its location and instructions to contact support with the file:

```
# log dump
creating /tmp/log dump.txt
The file /tmp/log dump.txt was created.  Please provide this file to
your support contact at ews-support@lairdtech.com
```

See Appendix X for an example of the log_dump.txt file

Setting the Kernel Log level

The log level is used by the kernel to determine the importance of a message and to decide whether it should be presented to the user immediately, by printing it console. The console will output kernel messages as they are put into the log if the loglevel is smaller (greater importance) then the console log level value. By default, the console kernel log contains messages of less then WARNING level (4). At run time, you can change the minimum level of the output.

To determine your current *console_loglevel*, enter the following into the command console:

```
# cat /proc/sys/kernel/printk
4       7       1       7
```

The first integer of the output shows your current console_loglevel; the second shows the default log level.

To change your current console_loglevel, write to this file and change the settings. In order to get all messages printed to the console, enter the following into the command prompt:

Americas: +1-800-492-2320
Europe: +44-1628-858-940
Hong Kong: +852 2923 0610

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

Once this is set, all kernel messages appear on the console.

Additional information on the kernel module loglevel can be found here:
https://github.com/torvalds/linux/blob/master/Documentation/kernel-parameters.txt#L2033

Setting the Driver and Supplicant Log Levels

The sdc_cli utility can set the logging level for both the driver and supplicant.

To see the current logging levels, issue:

```
# sdc cli logging show
    supplicant level: none
    driver level: none
```

To set a log level for the driver:

```
# sdc_cli logging set driver <0-3>
```

To set a level for the supplicant:

```
# sdc_cli logging set supplicant
```

UNDERSTANDING THE LOGS

syslog

For much of Linux, syslog is the common combined log. Most background programs (daemons) and other system scripts and programs write to the log. Any script can write to the log by using the program logger.

A typical line looks like the following:

```
Jan 2 16:53:58 summit user.notice client_wd(wlan0): scanning for udhcpc
```

It starts with a date and time stamp, followed by the hostname (the WB's default is ***summit***). This is followed by the type of message it is, in our example this is a user notice. Then is the application sending the message, followed by the message itself.

The formatting of the application and the message is up to the sender.

Common sources include:

- kernel – These are kernel printk messages (the same information as you'd get from dmesg)
- nis – Network interface status information
- laird[###] – Event messages from event_mon
- client_wd(interface) – dhcp watchdog monitor

Americas: +1-800-492-2320
Europe: +44-1628-858-940
Hong Kong: +852 2923 0610

Laird Events

The networking stack components all output event messages when certain things happen. This event system was designed so a network management application could be written to manage the state of the radio. A program, event_mon, runs on the WB and routes the event messages to the syslog or the console. When the WB boots, event_mon starts in the background and logs events to the syslog.

The following is an example of event_mon messages for a successful connection using LEAP authentication via a logread dump of syslog:

```
Jan 1 00:01:27 summit user.info laird[625]: Event: SDC_E_CONNECT_REQ Auth type: AUTH_OPEN
Jan 1 00:01:27 summit user.info laird[625]: Event: SDC_E_CONNECTION_STATE status:
ASSOCIATING
Jan 1 00:01:27 summit user.info laird[625]: Event: SDC_E_CONNECTION_STATE status:
ASSOCIATED
Jan 1 00:01:27 summit user.info laird[625]: AP Mac address: 54:78:1a:42:b0:d7
Jan 1 00:01:27 summit user.info laird[625]: Event: SDC_E_CONNECTION_STATE status:
AUTHENTICATING
Jan 1 00:01:27 summit user.info laird[625]: Auth reason: AUTH_REASON_UNSPEC
Jan 1 00:01:27 summit user.info laird[625]: AP Mac address: 54:78:1a:42:b0:d7
Jan 1 00:01:27 summit user.info laird[625]: Event: SDC_E_CONNECTION_STATE status:
AUTHENTICATED
Jan 1 00:01:27 summit user.info laird[625]: Auth reason: AUTH_REASON_UNSPEC
Jan 1 00:01:27 summit user.info laird[625]: AP Mac address: 54:78:1a:42:b0:d7
Jan 1 00:01:27 summit user.info laird[625]: Event: SDC_E_DHCP status: REQUESTING reason:
DHCP_REASON_UNSPEC
Jan 1 00:01:27 summit user.info laird[625]: AP Mac address: 54:78:1a:42:b0:d7
Jan 1 00:01:28 summit user.info laird[625]: Event: SDC_E_DHCP status: BOUND reason:
IP_ADDRESS_DIFFERENT
Jan 1 00:01:28 summit user.info laird[625]: interface: wlan0
Jan 1 00:01:28 summit user.info laird[625]: address: 10.1.44.114
Jan 1 00:01:28 summit user.info laird[625]: subnet_mask: 255.255.255.0
Jan 1 00:01:28 summit user.info laird[625]: routers: 10.1.44.1
Jan 1 00:01:28 summit user.info laird[625]: lease_time: 14400
Jan 1 00:01:28 summit user.info laird[625]: message_type: 5
Jan 1 00:01:28 summit user.info laird[625]: dns_servers: 10.1.44.2 10.1.44.32
Jan 1 00:01:28 summit user.info laird[625]: dhcp_server: 1.1.1.1
Jan 1 00:01:28 summit user.info laird[625]: domain_name: akron.com
Jan 1 00:01:28 summit user.info laird[625]: renew: 4 1970/01/01 02:01:28
Jan 1 00:01:28 summit user.info laird[625]: rebind: 4 1970/01/01 03:31:28
Jan 1 00:01:28 summit user.info laird[625]: expire: 4 1970/01/01 04:01:28
Jan 1 00:01:28 summit user.info laird[625]: AP Mac address: 54:78:1a:42:b0:d7
```

Below is an example of a failed connection attempt with LEAP using the wrong password:

Americas: +1-800-492-2320
Europe: +44-1628-858-940
Hong Kong: +852 2923 0610

```
Jan 1 00:03:56 summit user.info laird[625]: Event: SDC_E_CONNECT_REQ Auth type: AUTH_OPEN
Jan 1 00:03:56 summit user.info laird[625]: Event: SDC_E_CONNECTION_STATE status:
ASSOCIATING
Jan 1 00:03:56 summit user.info laird[625]: Event: SDC_E_CONNECTION_STATE status:
ASSOCIATED
Jan 1 00:03:56 summit user.info laird[625]: AP Mac address: 00:26:cb:f3:8c:09
Jan 1 00:03:56 summit user.info laird[625]: Event: SDC_E_CONNECTION_STATE status:
AUTHENTICATING
Jan 1 00:03:56 summit user.info laird[625]: Auth reason: AUTH_REASON_UNSPEC
Jan 1 00:03:56 summit user.info laird[625]: AP Mac address: 00:26:cb:f3:8c:09
Jan 1 00:03:57 summit user.info laird[625]: Event: SDC_E_CONNECTION_STATE status:
AUTH_ERROR
Jan 1 00:03:57 summit user.info laird[625]: Auth reason: INVALID_CREDENTIALS
Jan 1 00:03:57 summit user.info laird[625]:  AP Mac address: 00:26:cb:f3:8c:09
```

A typical event_mon logging message contains and Event, Status, Reason information. Once the client associates to an AP, the AP MAC address is then printed. For SDC_E_DHCP events that indicate a status of BOUND, RENEWED, DECONFIG, or RELEASED, an output of the DHCP leases file also displays.

Currently the event_mon running at start-up outputs SDC_E_CONNECTION_STATE, SDC_E_DHCP, SDC_E_READY, SDC_E_ROAM, SDC_E_SCAN_REQ, SDC_E_DISCONNECT_REQ, SDC_E_SCAN, SDC_E_REGDOMAIN, SDC_E_CMDERROR, SDC_E_INTERNAL, and SDC_E_FW_ERROR events to the syslog.

Note: For more about Laird Events, see the ***Events*** section of the *SDK Programmer's Guide.*

Kernel Log

The kernel log is read by the program dmesg. Use the parameter –c to clear the log after reading it.

All subsystems in the kernel use the log, printing messages to it. The messages are labelled with a priority and usually with the subsystem from which it came.

Typical log entries look like the following:

```
atmel_usba_udc 500000.gadget: FIFO at 0x00500000 mapped at c4c00000
 gadget: using random host ethernet address
usb0: MAC 7e:07:10:8c:e5:e5
usb0: HOST MAC 02:bb:09:17:5b:66
 gadget: Ethernet Gadget, version: Memorial Day 2008
 gadget: g_ether ready
ath6kl: INIT GENERIC NETLINK ATHEROS COM
ath6kl: ar6003 hw 2.1.1 sdio fw 3.4.0.0084\x01 api 4
```

Messages are pre-pended with the subsystem from which they arrive.

Important subsystems:

- **ath6kl** – This is the Wi-Fi chip driver for the WB45. If you're having trouble with the Wi-Fi chip specifically, these messages are key.
- **cfg80211** – This is the subsystem that is responsible for Wi-Fi configuration parameters in the kernel.

Americas: +1-800-492-2320
Europe: +44-1628-858-940
Hong Kong: +852 2923 0610

- **mmc0** – This is the MMC card and SDIO driver subsystem. Look here if you're having low-level SDIO problems.
- **UBIFS** – Our devices use the UBI filesystem. This is the tag for these drivers.

If a message doesn't make sense or is confusing, it's advisable to search for the message text online or search the Linux kernel source code for answers.

TROUBLESHOOTING AND DEBUGGING

Version Information

It's important to know how to find the version information from your device. This is one of the first pieces of information Laird Support and Engineering will request.

A single place for most important version information is sdc_cli. Enter **sdc_cli version** at the terminal and hit return:

```
# sdc_cli version
CLI: 3.5.0.5
SDK: 3.5.0.5
Hardware Chipset: 45
Driver: 3.5.0.1
Firmware: ar6003 hw 2.1.1 fw 3.4.0.0084 api 4
Supplicant: sdcsupp v3.5.20.6
Build: Laird Linux wb45n-laird_fips-3.5.0.17
```

This same information is available in the WebLCM interface on the About tab.

Another useful piece of information is the kernel version. Enter **uname –a** at the terminal and hit return:

```
# uname -a
 Linux summit 3.8.0-laird21.01 #1 PREEMPT Thu Aug 7 15:57:13 EDT 2014 armv5tejl
 GNU/Linux
```

Network Information

A large amount of information is presented by our various network programs and scripts. Following is a list of commands that provide useful information:

- **ifrc -v wlan0** – This gives you information about a specific interface (wlan0 one in this example).
- **ifrc –v** – Gives detailed information about all the interfaces.
- **ifrc br0** – Gives information about the bridge mode if enabled.
- **ip route show** – shows routing table information.
- **ip addr show** – shows the IP address information on configured interfaces.
- **cat /var/log/ifrc/wlan0** – shows the debug log for ifrc's management of the Wi-Fi interface.

The following is an example session for finding this information (commands in bold):

Americas: +1-800-492-2320
Europe: +44-1628-858-940
Hong Kong: +852 2923 0610

```
# ifrc -v wlan0
Configuration for interface: ...managed, active, associated

wlan0:   <BROADCAST,MULTICAST,UP,LOWER UP> mtu 1500 state UP
         link/ether 40:2c:f4:d3:9e:b5 brd ff:ff:ff:ff:ff:ff
         inet 192.168.10.102/24 brd 192.168.10.255 scope global wlan0

WiFi:    active
Associated to 64:e9:50:d1:0c:40 (on wlan0)
        SSID: ccopen1
        freq: 2427
        signal: -43 dBm
        tx bitrate: 72.2 MBit/s MCS 7 short GI

Routing:
192.168.10.0/24  proto kernel  scope link  src 192.168.10.102

ARP:
  ...
Connections:
  ...

Processes:
  345 S    /usr/bin/sdcsupp -iwlan0 -Dnl80211 -s
  500 S    ifplugd -iwlan0 -miff -s -fa -qMp -u0 -d0 -Ir/etc/network/ifrc.sh
  587 S    udhcpc -iwlan0 -R -t4 -T2 -A5 -b -o -Olease -Odomain -Odns -Ohostname -Osubnet
-Orouter -Oserverid -Obroadcast -p/var/run/dhclient.wlan0.pid -s/etc/dhcp/udhcpc.script
  589 S    ash /etc/dhcp/client wd -iwlan0

# ifrc -v
Configuration for all interfaces                              (try -h to see usage)
       Available, but not configured:  usb0

lo: <LOOPBACK,UP,LOWER UP> mtu 65536 noqueue state UNKNOWN
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo

eth0:    <BROADCAST,MULTICAST,UP,LOWER UP> mtu 1500 state UP
    link/ether 00:17:23:d0:12:e0 brd ff:ff:ff:ff:ff:ff
    inet 192.168.0.111/24 brd 192.168.0.255 scope global eth0

usb0:    <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 state DOWN
    link/ether 7e:07:10:8c:e5:e5 brd ff:ff:ff:ff:ff:ff
    inet 192.168.3.1/24 brd 192.168.3.255 scope global usb0

wlan0:   <BROADCAST,MULTICAST,UP,LOWER UP> mtu 1500 state UP
    link/ether 40:2c:f4:d3:9e:b5 brd ff:ff:ff:ff:ff:ff
    inet 192.168.10.102/24 brd 192.168.10.255 scope global wlan0

WiFi:    active
Associated to 64:e9:50:d1:0c:40 (on wlan0)
        SSID: ccopen1
        freq: 2427
        signal: -44 dBm
        tx bitrate: 72.2 MBit/s MCS 7 short GI

Routing:
default
    nexthop via 192.168.10.2  dev wlan0 weight 4
    nexthop via 192.168.0.1  dev eth0 weight 1
192.168.0.0/24 dev eth0  proto kernel  scope link  src 192.168.0.111
192.168.3.0/24 dev usb0  proto kernel  scope link  src 192.168.3.1
192.168.10.0/24 dev wlan0  proto kernel  scope link  src 192.168.10.102

DNS:    resolv.conf
nameserver 68.87.76.182
nameserver 8.8.8.8
nameserver 8.8.8.8
options timeout:3
```


Americas: +1-800-492-2320
Europe: +44-1628-858-940
Hong Kong: +852 2923 0610

```
# ifrc br0
  ...not available, not a kernel-resident interface
# ip route show
default
    nexthop via 192.168.10.2  dev wlan0 weight 4
    nexthop via 192.168.0.1  dev eth0 weight 1
192.168.0.0/24 dev eth0  proto kernel  scope link  src 192.168.0.111
192.168.3.0/24 dev usb0  proto kernel  scope link  src 192.168.3.1
192.168.10.0/24 dev wlan0  proto kernel  scope link  src 192.168.10.102
# ip addr show
1: lo: <LOOPBACK,UP,LOWER UP> mtu 65536 qdisc noqueue state UNKNOWN
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
    inet6 ::1/128 scope host
       valid lft forever preferred lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER UP> mtu 1500 qdisc pfifo fast state UP qlen 1000
    link/ether 00:17:23:d0:12:e0 brd ff:ff:ff:ff:ff:ff
    inet 192.168.0.111/24 brd 192.168.0.255 scope global eth0
    inet6 fe80::217:23ff:fed0:12e0/64 scope link
       valid lft forever preferred lft forever
3: usb0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc pfifo fast state DOWN qlen
1000
    link/ether 7e:07:10:8c:e5:e5 brd ff:ff:ff:ff:ff:ff
    inet 192.168.3.1/24 brd 192.168.3.255 scope global usb0
4: wlan0: <BROADCAST,MULTICAST,UP,LOWER UP> mtu 1500 qdisc pfifo fast state UP qlen 1000
    link/ether 40:2c:f4:d3:9e:b5 brd ff:ff:ff:ff:ff:ff
    inet 192.168.10.102/24 brd 192.168.10.255 scope global wlan0
    inet6 fe80::422c:f4ff:fed3:9eb5/64 scope link
       valid lft forever preferred lft forever
# cat /var/log/ifrc/wlan0
mp cdt:dhcp
eni sk:2124956927 349
ifrc i:wl=wlan0 fls="" mm= vm= qm=>/dev/null
     -- v20140612 - md5:fffbae3fc65c1514a59cda208796287f
        11.15   /etc/network/ifrc.sh wl up  -- 110 rcS
        12.57   /etc/dhcp/udhcpc.script deconfig
        12.80   /etc/dhcp/udhcpc.script requesting
        13.00   /etc/dhcp/udhcpc.script bound

      1620.89   /sbin/ifrc -v wlan0

      2401.74   /sbin/ifrc -v wlan0
#
```

Wi-Fi Protocol Information

The above examples provide a great amount of information about the Wi-Fi interface. The following tools illustrate how to troubleshoot the Wi-Fi interface.

Profile Information

Wi-Fi connections are managed by profiles. To troubleshoot a profile, first look in the WB's profile settings. The WebLCM presents this information and allows you to configure it on the Wi-Fi tab (Figure 2):

Americas: +1-800-492-2320
Europe: +44-1628-858-940
Hong Kong: +852 2923 0610

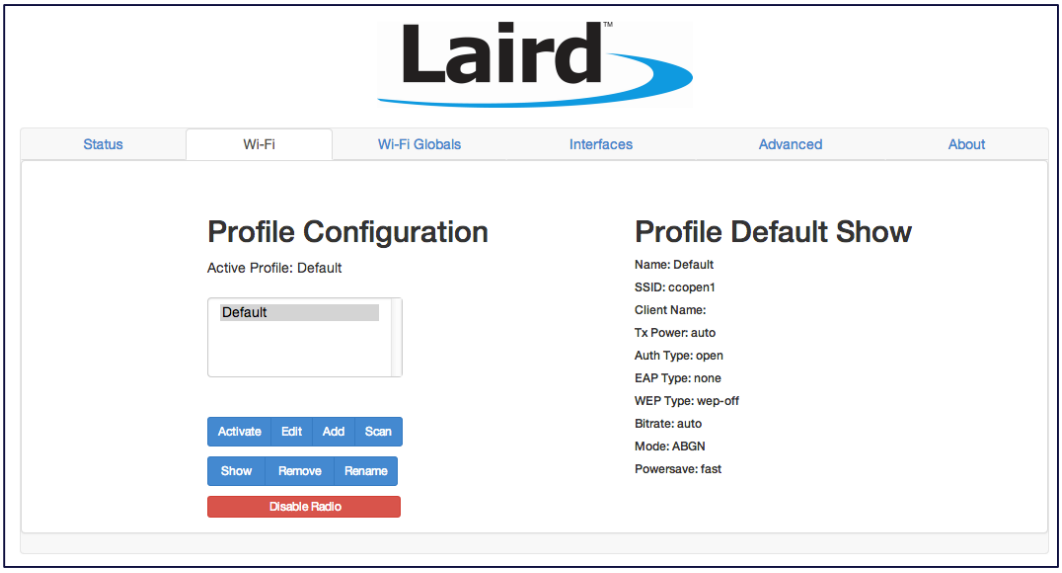

Figure 2: Wi-Fi tab of WebLCM

The window above shows the information after selecting the Default profile and clicking **Show**.

You can accomplish the same task with the sdc_cli program on the console:

```
# sdc cli profile list
 Default ACTIVE

# sdc cli profile Default show
  Name: Default
  SSID: ccopen1
  Client Name:
  Tx Power: auto
  Auth Type: open
  EAP Type: none
  WEP Type: wep-off
  Bitrate: auto
  Mode: ABGN
  Powersave: fast
```

When the WB is having trouble connecting to an AP, the most likely problem is some overlooked configuration problem. See the *sdc_cli User Guide* for more information.

Atheros Wi-Fi Drive Debugging Logs

The driver logging level can be set through the sdc_cli as indicated in the Setting the Driver and Supplicant Log Levels section of this document.

This level can also be set manually and for specific flags. The ath6kl driver for the AR6003 and AR6004 chips used on the WBs can output much more specific information.

The debugging flags are values set as a bitmask in drivers/net/wireless/ath/ath6kl/debug.h:

Americas: +1-800-492-2320
Europe: +44-1628-858-940
Hong Kong: +852 2923 0610

```
enum ATH6K DEBUG MASK {
        ATH6KL DBG CREDIT  = BIT(0),
        /* hole */
        ATH6KL DBG WLAN TX      = BIT(2),     /* wlan tx */
        ATH6KL DBG WLAN RX      = BIT(3),     /* wlan rx */
        ATH6KL DBG BMI          = BIT(4),     /* bmi tracing */
        ATH6KL DBG HTC          = BIT(5),
        ATH6KL DBG HIF          = BIT(6),
        ATH6KL DBG IRQ          = BIT(7),     /* interrupt processing */
        /* hole */
        /* hole */
        ATH6KL DBG WMI          = BIT(10),    /* wmi tracing */
        ATH6KL DBG TRC             = BIT(11),    /* generic func tracing */
        ATH6KL DBG SCATTER = BIT(12),    /* hif scatter tracing */
        ATH6KL DBG WLAN CFG     = BIT(13),    /* cfg80211 i/f file tracing */
        ATH6KL DBG RAW BYTES    = BIT(14),    /* dump tx/rx frames */
        ATH6KL DBG AGGR         = BIT(15),    /* aggregation */
        ATH6KL DBG SDIO         = BIT(16),
        ATH6KL DBG SDIO DUMP    = BIT(17),
        ATH6KL DBG BOOT         = BIT(18),    /* driver init and fw boot */
        ATH6KL DBG WMI DUMP     = BIT(19),
        ATH6KL DBG SUSPEND = BIT(20),
        ATH6KL DBG USB          = BIT(21),
        ATH6KL DBG USB BULK     = BIT(22),
        ATH6KL DBG RECOVERY     = BIT(23),
        ATH6KL DBG ANY             = 0xffffffff  /* enable all logs */
};
```

Combining the different bits enables varying debug output from the driver.

The debug_mask module parameter may be set either at the time you load the driver or during run time. Certain bits, like the BMI tracing and BOOT flags, are only useful when set at the time that the driver is loaded.

The most useful flag to set when debugging is the WMI flag. Laird support engineers might ask you to enable this bit when helping to track down a problem.

To set the debug mask on load:

```
# modprobe ath6kl core debug mask=0x00050000
# modprobe ath6kl_sdio
```

To set the debug mask at runtime:

```
# echo 0x00002400 > /sys/module/ath6kl_core/parameters/debug_mask
```

Supplicant Logs

As with the driver, the supplicant logging level can be set via the sdc_cli utility. See the Setting the Driver and Supplicant Log Levels section of this document. .

Roaming Information

The AR6003/AR6004 of the WB45NBT/WB50NBT allows you to dump the roam table in order to facilitate debugging of roam situations. To look at the table, you must mount the debugfs, which is not mounted by default:

```
# mount -t debugfs nodev /sys/kernel/debug
```


Americas: +1-800-492-2320
Europe: +44-1628-858-940
Hong Kong: +852 2923 0610

Then the roam table becomes available:

```
# cat /sys/kernel/debug/ieee80211/phy0/ath6kl/roam_table
roam_mode=0

# roam_util bssid rssi rssidt last_rssi util bias
510 64:e9:50:d1:0c:40 34 0 34 15 0
```

The roam table lists, with one candidate per line, the different candidate APs the AR6003/AR6004 might consider roaming to. The different parameters are, from left to right:

- **roam_util** – This is the computed likelihood the chip chooses this AP if it roams. The entry with the highest roam_util is chosen. The majority of the roam_util is calculated as follows:
  *(roam_util = rssi * util + bias)*
  In other words, it is calculated by taking [RSSI field] times [util field] and, if enabled, the bias is added.
- **BSSID** – The BSSID of the AP
- **RSSI** – An average of the last several RSSI measurements.
- **RSSIDT** – The rate of change of the last several RSSI averages calculated for this table.
- **last_rssi** – The last RSSI reading taken.
- **util** – A specific utility multiplier which is either 15 for the currently connected AP or 10 for non-connected APs.
- **bias** – If set, and the bias roam mode is turned on, this value is added into the roam_util equation.

Wi-Fi Firmware Restart

Laird's AR6003 and AR6004 firmware and driver on the WB is capable of reloading and restarting in the event of an unrecoverable error. By default, this feature is enabled the scripts that load and start Laird's drivers.

If you need to enable the feature when manually loading the drivers, you simply enable the correct parameters: *recovery_enable =1 heart_beat_poll=200* parameters:

```
# modprobe ath6kl_core recovery_enable=1 heart_beat_poll=200
# modprobe ath6kl_sdio
```

Note: The recovery enable feature is already enabled by default in GA3 and higher releases. It is unavailable in earlier releases.

Wi-Fi Protocol Capture

Using a Wi-Fi protocol sniffer like Wireshark is helpful in debugging protocol problems. Laird support may ask you for a capture in PCAP format. For more information, visit https://www.wireshark.org/.

Flash Image

Information about the current running environment and what images are flashed on which partition can sometimes be useful in troubleshooting problems. The fw_select and fw_printenv utilities can be used to gather this information.

Americas: +1-800-492-2320
Europe: +44-1628-858-940
Hong Kong: +852 2923 0610

WB45

```
# fw select --show
wb45n - flash memory
  mtd     address      size       description
    0:   0x00000000   0x00020000  at91bs        3.4.4-laird2
    1:   0x00020000   0x00080000  u-boot        2013.01-laird2-00096-ge5cf952-dirty
    2:   0x000a0000   0x00020000  u-boot-env
    3:   0x000c0000   0x00020000  redund-env
 -> 4:   0x000e0000   0x00280000  kernel-a      Linux-3.8.0-laird21.01
    5:   0x00360000   0x00280000  kernel-b      Linux-3.8.0-laird21.01
 -> 6:   0x005e0000   0x02600000  rootfs-a      wb45n-laird fips-3.5.0.17
    7:   0x02be0000   0x02600000  rootfs-b      wb45n-laird fips-3.5.0.17
    8:   0x051e0000   0x02d20000  user
    9:   0x07f00000   0x00080000  logs

Laird Linux wb45n-laird fips-3.5.0.17
 -> summit 3.8.0-laird21.01 on rootfs-a

bootup:
  kernel-a
  rootfs-a
```

WB50

```
# fw select --show
wb50n - flash memory
  mtd     address      size       description
    0:   0x00000000   0x00020000  at91bs        3.7.1-laird05
    1:   0x00020000   0x00080000  u-boot        2014.07-laird03
    2:   0x000a0000   0x00020000  u-boot-env
    3:   0x000c0000   0x00020000  u-boot-env
    4:   0x000e0000   0x00500000  kernel-a      4.1.13-laird12
 -> 5:   0x005e0000   0x00500000  kernel-b      4.1.13-laird12
    6:   0x00ae0000   0x03000000  rootfs-a      wb50n-laird-3.5.2.16
 -> 7:   0x03ae0000   0x03000000  rootfs-b      wb50n-laird-3.5.2.16
    8:   0x06ae0000   0x014e0000  user

Laird Linux wb50n-laird-3.5.2.16
 -> summit 4.1.13-laird12 on rootfs-b

bootup:
  kernel-b
  rootfs-b

# fw printenv
RegDomain=3
 mtd=mtdparts default; set bootargs ${bootargs} ${wbti} ${mtdparts}
autoload=no
baudrate=115200
bootdelay=1
ethact=macb0
ethaddr=00:17:23:d0:12:e0
fileaddr=22000000
filesize=1680000
gatewayip=192.168.0.1
ipaddr=192.168.0.60
serverip=192.168.0.18
stderr=serial
stdin=serial
stdout=serial
bootargs=console=ttyS0,115200 loglevel=4 rw noinitrd mem=64M rootfstype=ubifs
root=ubi0:rootfs ubi.mtd=6
bootcmd=nand read 0x22000000 0x000e0000 0x00280000; run _mtd; bootm
```


Americas: +1-800-492-2320
Europe: +44-1628-858-940
Hong Kong: +852 2923 0610

APPENDIX A: ATH6KL DEBUG FLAGS TABLE

Print the following table and use it as a handy tool to calculate the debug log mask values for the ath6kl driver. Either set the debug value on the command line when starting the driver or set the value in /sys/module/ath6kl_core/parameters/debug_mask at run time.

31	30	29	28	27	26	25	24	RECOVERY 23	USB_BULK 22	USB 21	SUSPEND 20	WMI_DUMP 19	BOOT 18	SDIO_DUMP 17	SDIO 16	AGGR 15	RAW_BYTES 14	WLAN_CFG 13	SCATTER 12	TRC 11	WMI 10	9	8	IRQ 7	HIF 6	HTC 5	BMI 4	WLAN_RX 3	WLAN_TX 2	1	CREDIT 0

APPENDIX X: EXAMPLE LOG_DUMP.TXT FILE

```
= = = = = = = = = = = = = = = = = = = =
log_dump file created with version  20151125
Thu Jan  1 07:12:19 UTC 1970
-----uptime-----
 07:12:19 up  7:12,  load average: 1.20, 1.24, 1.23
-----system-----
Laird Linux wb50n-laird-3.5.2.16
-----kernel info-----
Linux summit 4.1.13-laird12 #1 PREEMPT Thu Mar 24 23:37:58 EDT 2016 armv7l GNU/L
inux
-----kernel cmd line-----
console=ttyS0,115200 earlyprintk loglevel=4 mem=64M rw rootfstype=ubifs ubi.mtd=
6 root=ubi0:rootfs
-----uboot env-----
autoload=no
autostart=no
baudrate=115200
bootdelay=3
dnsip=168.95.192.1
```


Americas: +1-800-492-2320
Europe: +44-1628-858-940
Hong Kong: +852 2923 0610

```
ethact=macb0
fileaddr=20000000
filesize=1940000
gatewayip=192.168.100.1
ipaddr=192.168.100.219
loadaddr=0x22000000
lrd_name=sama5d31-wb50n
netmask=255.255.255.0
serverip=192.168.100.121
stderr=serial
stdin=serial
stdout=serial
RegDomain=3
md5_at91bs=0ad0e8e6960eee6e62695fe91bfb94bb
md5_u-boot=4672abc7a4c9e288bcf4bfd09832d834
ethaddr=c0:ee:40:00:50:00
md5_kernel-b=87c37a75288af259506834b8727357d3
md5_rootfs-b=71df71a81216b1a0088c3e332b277c91
md5_kernel-a=b89ed7fbd22f65095a22bb09b330281d
md5_rootfs-a=ef9ff54a6f23bbc1e4293dfad4477afb
bootargs=rw rootfstype=ubifs ubi.mtd=6 root=ubi0:rootfs
bootcmd=nand read 0x22000000 0x000e0000 0x00500000; bootm
-----sdc_cli version-----
CLI: 3.5.2.10
SDK: 3.5.3.11
Hardware Chipset: 50 Workgroup Bridge
Driver: 3.5.0.1
Firmware: ar6004 hw 3.0 fw 3.5.0.10008 api 5
Supplicant: sdcsupp v40.3.3.0
Build: Laird Linux wb50n-laird-3.5.2.16
-----sdc_cli status-----
    Status: Authenticated
    Profile name: bill
    SSID: Bill Wi The Science Fi
    Channel: 11
    RSSI: -29
    Device Name:
    MAC: c0:ee:40:0a:b6:99
    IP: 192.168.1.1 (Static)
```


Americas: +1-800-492-2320
Europe: +44-1628-858-940
Hong Kong: +852 2923 0610

```
    IPv6: fe80::c2ee:40ff:fe0a:b699
    AP MAC: 54:a0:50:d4:60:92
    Bit Rate: 144.4 Mbps
    Tx Power: 25 mW
    Beacon Period: 100 ms
    DTIM: 1
-----sdc_cli profile list-----
 Default
 bill ACTIVE
 foo
-ap1
 ap4
- denotes auto-profile (auto-profile is Inactive)

2 profiles
-----------
Profile 1:
    Name: Default
    SSID:
    Client Name:
    Tx Power: Auto
    Auth Type: Open
    EAP Type: None
    WEP Type: None
    Mode: ABGN
    Powersave: Fast
    PSP Delay: 200 msec
--------------------
Profile 2:
    Name: bill ACTIVE
    SSID: Bill Wi The Science Fi
    Client Name:
    Tx Power: Auto
    Auth Type: Open
    EAP Type: None
    WEP Type: WPA-PSK-TKIP
    Mode: ABGN
    Powersave: Fast
    PSP Delay: 200 msec
```


Americas: +1-800-492-2320
Europe: +44-1628-858-940
Hong Kong: +852 2923 0610

```
    PSK: ********
--------------------


-----ifrc -v wlan0-----
Configuration for interface: wlan0 - static/managed, active, associated


wlan0:  <BROADCAST,MULTICAST,UP,LOWER_UP>
         mtu 1500 group default state UP
        link/ether c0:ee:40:0a:b6:99 brd ff:ff:ff:ff:ff:ff
        inet 192.168.1.1/30 brd 192.168.1.255 scope global
        inet6 fe80::c2ee:40ff:fe0a:b699/64 scope link

WiFi:   active
Associated to 54:a0:50:d4:60:92 (on wlan0)
        SSID: Bill Wi The Science Fi
        freq: 2462
        signal: -29 dBm
        tx bitrate: 144.4 MBit/s MCS 15 short GI


Routing:


ARP:
  ...M192.168.2.39 at 10:40:f3:95:9f:fa STALE


Processes:
 1287 S    /usr/bin/sdcsupp -iwlan0 -Dnl80211 -s
 1345 S    ifplugd -iwlan0 -miff -s -fa -qMp -u0 -d0 -Ir/usr/sbin/ifrc -x&
-----ifrc -v-----
Configuration for all interfaces:                          (try -h to see usage
)


lo:     <LOOPBACK,UP,LOWER_UP>
         mtu 65536 noqueue group default  state UNKNOWN
        link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
        inet 127.0.0.1/8 scope host
```


Americas: +1-800-492-2320
Europe: +44-1628-858-940
Hong Kong: +852 2923 0610

```
        inet6 ::1/128 scope host

eth0:   <BROADCAST,MULTICAST,UP,LOWER_UP>
         mtu 1500 group default state UP
        link/ether c0:ee:40:00:50:00 brd ff:ff:ff:ff:ff:ff
        inet 192.168.2.124/24 brd 192.168.2.255 scope global
        inet6 fe80::c2ee:40ff:fe00:5000/64 scope link

usb0:   <NO-CARRIER,BROADCAST,MULTICAST,UP>
         mtu 1500 group default state DOWN
        link/ether b2:6f:1e:49:cc:9c brd ff:ff:ff:ff:ff:ff
        inet 192.168.3.1/24 brd 192.168.3.255 scope global

wlan0:  <BROADCAST,MULTICAST,UP,LOWER_UP>
         mtu 1500 group default state UP
        link/ether c0:ee:40:0a:b6:99 brd ff:ff:ff:ff:ff:ff
        inet 192.168.1.1/30 brd 192.168.1.255 scope global
        inet6 fe80::c2ee:40ff:fe0a:b699/64 scope link

WiFi:   active
Associated to 54:a0:50:d4:60:92 (on wlan0)
        SSID: Bill Wi The Science Fi
        freq: 2462
        signal: -29 dBm
        tx bitrate: 144.4 MBit/s MCS 15 short GI


Routing:
default via 192.168.2.1 dev eth0
192.168.1.0/30 dev wlan0  proto kernel  scope link  src 192.168.1.1
192.168.2.0/24 dev eth0  proto kernel  scope link  src 192.168.2.124
192.168.3.0/24 dev usb0  proto kernel  scope link  src 192.168.3.1


Rules..
0:      from all lookup local
32764:  from all to 192.168.2.124 lookup t.eth0
32765:  from 192.168.2.124 lookup t.eth0
32766:  from all lookup main
32767:  from all lookup default
```


Americas: +1-800-492-2320
Europe: +44-1628-858-940
Hong Kong: +852 2923 0610

```
DNS:M   resolv.conf
nameserver 192.168.2.1
nameserver 8.8.8.8
options timeout:3
-----bridge staus------
  ...not available, not a kernel-resident interface
----image info-----
wb50n - flash memory
  mtd     address     size      description
    0:  0x00000000  0x00020000  at91bs       3.7.1-laird05
    1:  0x00020000  0x00080000  u-boot       2014.07-laird03
    2:  0x000a0000  0x00020000  u-boot-env
    3:  0x000c0000  0x00020000  u-boot-env
 -> 4:  0x000e0000  0x00500000  kernel-a     4.1.13-laird12
    5:  0x005e0000  0x00500000  kernel-b     4.1.13-laird09
 -> 6:  0x00ae0000  0x03000000  rootfs-a     wb50n-laird-3.5.2.16
    7:  0x03ae0000  0x03000000  rootfs-b     20160314
    8:  0x06ae0000  0x014e0000  user


wb50n - MLaird Linux wb50n-laird-3.5.2.16
 -> summit 4.1.13-laird12 on rootfs-a


bootup:
  kernel-a
  rootfs-a
-----route info-----
default via 192.168.2.1 dev eth0
192.168.1.0/30 dev wlan0  proto kernel  scope link  src 192.168.1.1
192.168.2.0/24 dev eth0  proto kernel  scope link  src 192.168.2.124
192.168.3.0/24 dev usb0  proto kernel  scope link  src 192.168.3.1
-----ifrc log-----
mp_cdt:f4=static ip=192.168.1.1 bc=192.168.1.255 nm=255.255.255.252 cdt{ post_c
fg_do='/etc/network/dsrc-rules.conf' pre_dcfg_do='/etc/network/dsrc-rules.conf'
; }
eni_sk:1466017293 324
ifrc_i:wlan0=wlan0 fls="" mm= vm= qm=; mpr=yes
```


Americas: +1-800-492-2320
Europe: +44-1628-858-940
Hong Kong: +852 2923 0610

```
    -- v2.03
        14.18 -- 492_ifrc wlan0 up <- rcS
  post-cfg-do: dsrc-rules.conf 127


     25110.02 -- 1168_ifrc wlan0 restart


     25110.40 -- 1189_ifrc -x wlan0 down
  pre-dcfg-do: dsrc-rules.conf 127


     25114.72 -- 1303_ifrc wlan0 up
  post-cfg-do: dsrc-rules.conf 127


     25116.45 -- 1368_ifrc wlan0 up <- ifplugd
  dt->up wlan0 up f4=static
  probable roam, was dormant

     25940.36 -- 1712_ifrc -v wlan0
-----top-----
 Mem: 26940K used, 33020K free, 704K shrd, 0K buff, 12576K cached
CPU:  23% usr   0% sys   0% nic  76% idle   0% io   0% irq   0% sirq
Load average: 1.61 1.33 1.26 1/59 1876
  PID  PPID USER     STAT   VSZ %VSZ %CPU COMMAND
  679   677 root     S     8080  13%   8% /usr/bin/php-cgi
 1301     1 root     S    48872  81%   0% /usr/bin/event_mon -ologging -b0x0000
003FA3008000 -m
  545     1 root     S    14172  24%   0% smartSS -E 7 -d -a -c /usr/share/smar
tBASIC/apps/$autorun$.SPPBridge.Socket.wb.sb
 1287     1 root     S    14000  23%   0% /usr/bin/sdcsupp -iwlan0 -Dnl80211 -s


  677     1 root     S     5112   9%   0% /usr/sbin/lighttpd -f /etc/lighttpd/l
ighttpd.conf
   88     1 root     S     2968   5%   0% /sbin/syslogd -D -C512 -m0
   95     1 root     S     2728   5%   0% /sbin/udevd -d
  555     1 root     S     2536   4%   0% -sh
  915     1 root     S     2532   4%   0% /usr/sbin/inetd -e /tmp/inetd.conf
 1690   555 root     S     2460   4%   0% {log_dump} /bin/sh /usr/bin/log_dump
```


Americas: +1-800-492-2320
Europe: +44-1628-858-940
Hong Kong: +852 2923 0610

```
    1     0 root     S     2456   4%   0% init
  267     1 root     S     2456   4%   0% {lighty_wd} /bin/ash /usr/sbin/lighty
_wd
  627     1 root     S     2456   4%   0% ash /etc/dhcp/client_wd -ieth0
 1876  1690 root     R     2456   4%   0% top -b -n 1
   90     1 root     S     2456   4%   0% /sbin/klogd
  339     1 root     S     2456   4%   0% ifplugd -ieth0 -s -fa -qMp -u0 -d0 -I
r/usr/sbin/ifrc -x&
  344     1 root     S     2456   4%   0% ifplugd -iusb0 -s -fa -qMp -u0 -d0 -I
r/usr/sbin/ifrc -x&
  625     1 root     S     2456   4%   0% udhcpc -ieth0 -R -t4 -T2 -A5 -b -o -O
lease -Odomain -Odns -Ohostname -Osubnet -Orouter -Oserverid -Obroadcast -p/var
/run/dhclient.eth0.pid -s/etc/dhcp/udhcpc.script
 1345     1 root     S     2456   4%   0% ifplugd -iwlan0 -miff -s -fa -qMp -u0
 -d0 -Ir/usr/sbin/ifrc -x&
  553     1 root     S     1688   3%   0% input-event-daemon
  697     2 root     SW       0   0%   0% [kworker/0:2]
 1281     2 root     SW       0   0%   0% [ksdioirqd/mmc1]
    3     2 root     SW       0   0%   0% [ksoftirqd/0]
 1156     2 root     SW       0   0%   0% [kworker/u2:1]
    2     0 root     SW       0   0%   0% [kthreadd]
    7     2 root     SW       0   0%   0% [rcu_preempt]
   12     2 root     SW       0   0%   0% [kdevtmpfs]
 1450     2 root     SW       0   0%   0% [kworker/u2:2]
    5     2 root     SW<      0   0%   0% [kworker/0:0H]
    8     2 root     SW       0   0%   0% [rcu_sched]
    9     2 root     SW       0   0%   0% [rcu_bh]
   10     2 root     SW       0   0%   0% [watchdog/0]
   11     2 root     SW<      0   0%   0% [khelper]
   13     2 root     SW       0   0%   0% [khungtaskd]
   14     2 root     SW<      0   0%   0% [writeback]
   15     2 root     SW<      0   0%   0% [crypto]
   16     2 root     SW<      0   0%   0% [bioset]
   17     2 root     SW<      0   0%   0% [kblockd]
   19     2 root     SW       0   0%   0% [kswapd0]
   20     2 root     SW       0   0%   0% [fsnotify_mark]
   62     2 root     SW<      0   0%   0% [deferwq]
   64     2 root     SW       0   0%   0% [ubi_bgt0d]
   65     2 root     SW       0   0%   0% [ubifs_bgt0_0]
```


Americas: +1-800-492-2320
Europe: +44-1628-858-940
Hong Kong: +852 2923 0610

```
  166     2 root     SW       0   0%   0% [irq/94-atmel_us]
  351     2 root     SW<      0   0%   0% [cfg80211]
  663     2 root     SW<      0   0%   0% [ipv6_addrconf]
  706     2 root     SW       0   0%   0% [kworker/0:0]
 1271     2 root     SW<      0   0%   0% [ath6kl]
 1417     2 root     SW       0   0%   0% [kworker/u2:0]
-----df-----
Filesystem           1K-blocks      Used Available Use% Mounted on
ubi0:rootfs              39336     22252     17084  57% /
devtmpfs                 21712         0     21712   0% /dev
tmpfs                    29980         0     29980   0% /dev/shm
tmpfs                    29980       196     29784   1% /tmp
tmpfs                    29980       196     29784   1% /tmp
-----logread-----
(logread contents)
------processes------
PID   USER     COMMAND
    1 root     init
    2 root     [kthreadd]
    3 root     [ksoftirqd/0]
    5 root     [kworker/0:0H]
    7 root     [rcu_preempt]
    8 root     [rcu_sched]
    9 root     [rcu_bh]
   10 root     [watchdog/0]
   11 root     [khelper]
   12 root     [kdevtmpfs]
   13 root     [khungtaskd]
   14 root     [writeback]
   15 root     [crypto]
   16 root     [bioset]
   17 root     [kblockd]
   19 root     [kswapd0]
   20 root     [fsnotify_mark]
   62 root     [deferwq]
   64 root     [ubi_bgt0d]
   65 root     [ubifs_bgt0_0]
   88 root     /sbin/syslogd -D -C512 -m0
   90 root     /sbin/klogd
```


Americas: +1-800-492-2320
Europe: +44-1628-858-940
Hong Kong: +852 2923 0610

```
   95 root     /sbin/udevd -d
  166 root     [irq/94-atmel_us]
  267 root     {lighty_wd} /bin/ash /usr/sbin/lighty_wd
  339 root     ifplugd -ieth0 -s -fa -qMp -u0 -d0 -Ir/usr/sbin/ifrc -x&
  344 root     ifplugd -iusb0 -s -fa -qMp -u0 -d0 -Ir/usr/sbin/ifrc -x&
  351 root     [cfg80211]
  545 root     smartSS -E 7 -d -a -c /usr/share/smartBASIC/apps/$autorun$.SPPBr
idge.Socket.wb.sb
  553 root     input-event-daemon
  555 root     -sh
  625 root     udhcpc -ieth0 -R -t4 -T2 -A5 -b -o -Olease -Odomain -Odns -Ohost
name -Osubnet -Orouter -Oserverid -Obroadcast -p/var/run/dhclient.eth0.pid -s/e
tc/dhcp/udhcpc.script
  627 root     ash /etc/dhcp/client_wd -ieth0
  663 root     [ipv6_addrconf]
  677 root     /usr/sbin/lighttpd -f /etc/lighttpd/lighttpd.conf
  679 root     /usr/bin/php-cgi
  697 root     [kworker/0:2]
  706 root     [kworker/0:0]
  915 root     /usr/sbin/inetd -e /tmp/inetd.conf
 1156 root     [kworker/u2:1]
 1271 root     [ath6kl]
 1281 root     [ksdioirqd/mmc1]
 1287 root     /usr/bin/sdcsupp -iwlan0 -Dnl80211 -s
 1301 root     /usr/bin/event_mon -ologging -b0x0000003FA3008000 -m
 1345 root     ifplugd -iwlan0 -miff -s -fa -qMp -u0 -d0 -Ir/usr/sbin/ifrc -x&
 1417 root     [kworker/u2:0]
 1450 root     [kworker/u2:2]
 1690 root     {log_dump} /bin/sh /usr/bin/log_dump
 1879 root     ps
- - - - - - - - - - DMESG - - - - - - - - - -
[    0.000000] Booting Linux on physical CPU 0x0
...
```


Americas: +1-800-492-2320
Europe: +44-1628-858-940
Hong Kong: +852 2923 0610

REVISION HISTORY

Version	Date	Notes	Approver
1.0	06 April 2016	Initial Release	Jay White