SmartMesh WirelessHART Mote CLI Guide

Table of Contents

1 About This Guide

1.1 Related Documents

The following documents are available for the SmartMesh WirelessHART network:

Getting Started with a Starter Kit

- SmartMesh WirelessHART Easy Start Guide - walks you through basic installation and a few tests to make sure your network is working
- SmartMesh WirelessHART Tools Guide - the Installation section contains instructions for the installing the serial drivers and example programs used in the Easy Start Guide and other tutorials.

User Guide

- SmartMesh WirelessHART User's Guide - describes network concepts, and discusses how to drive mote and manager APIs to perform specific tasks, e.g. to send data or collect statistics. This document provides context for the API guides.

Interfaces for Interaction with a Device

- SmartMesh WirelessHART Manager CLI Guide - used for human interaction with a Manager (e.g. during development of a client, or for troubleshooting). This document covers connecting to the CLI and its command set.
- SmartMesh WirelessHART Manager API Guide - used for programmatic interaction with a manager. This document covers connecting to the API and its command set.
- SmartMesh WirelessHART Mote CLI Guide - used for human interaction with a mote (e.g. during development of a sensor applicaition, or for troubleshooting). This document covers connecting to the CLI and its command set.
- SmartMesh WirelessHART Mote API Guide - used for programmatic interaction with a mote. This document covers connecting to the API and its command set.

Software Development Tools

- SmartMesh WirelessHART Tools Guide - describes the various evaluation and development support tools included in the SmartMesh SDK including tools for exercising mote and manager APIs and visualizing the network.

Application Notes

- SmartMesh WirelessHART Application Notes - app notes covering a wide range of topics specific to SmartMesh WirelessHART networks and topics that apply to SmartMesh networks in general.

Documents Useful When Starting a New Design

- The Datasheet for the LTC5800-WHM SoC, or one of the castellated modules based on it, or the backwards compatible LTP5900 22-pin module.
- The Datasheet for the LTP5903-WHR embedded manager.
- A Hardware Integration Guide for the mote SoC or castellated module, or the 22-pin module - this discusses best practices for integrating the SoC or module into your design.
- A Hardware Integration Guide for the embedded manager - this discusses best practices for integrating the embedded manager into your design.
- A Board Specific Integration Guide - For SoC motes and Managers. Discusses how to set default IO configuration and crystal calibration information via a "fuse table".
- Hardware Integration Application Notes - contains an SoC design checklist, antenna selection guide, etc.
- The ESP Programmer Guide - a guide to the DC9010 Programmer Board and ESP software used to program firmware on a device.
- ESP software - used to program firmware images onto a mote or module.
- Fuse Table software - used to construct the fuse table as discussed in the Board Specific Integration Guide.

Other Useful Documents

- A glossary of wireless networking terms used in SmartMesh documentation can be found in the SmartMesh WirelessHART User's Guide.
- A list of Frequently Asked Questions

1.2 Conventions Used

The following conventions are used in this document:

`Computer type` indicates information that you enter, such as specifying a URL.

Bold type indicates buttons, fields, menu commands, and device states and modes.

Italic type is used to introduce a new term, and to refer to APIs and their parameters.

Tips provide useful information about the product.

Informational text provides additional information for background and context

Notes provide more detailed information about concepts.

Warning! Warnings advise you about actions that may cause loss of data, physical harm to the hardware or your person.

```
code blocks display examples of code
```

The CLI commands are described using the following notations and terminology:

|	Indicates alternatives for a field. For example, *<moteId>* | *#<MAC>* indicates that you can specify a mote by its mote ID or MAC address.
< >	Indicates a required field.
{ }	Indicates a group of fields.
[ ]	Indicates an optional field.
MAC address	When specifying a MAC address, do not use spaces. You may omit leading zeros and hyphens. In cases where the command syntax allows either the MAC address or mote ID to be specified, the MAC address must be preceded by the # symbol. The following examples are all valid: 22CA 00000000000022CA 00-00-00-00-00-00-22-CA

1.3 Revision History

Revision	Date	Description
1	07/12/2012	Initial Release
2	07/31/2012	Added mxtal command
3	03/18/2013	Numerous small changes
4	10/22/2013	Added nwl, compliandMode commands; Corrected power source parameter
5	04/04/2014	Clarified description of zeroize;
6	10/28/2014	Fixed mset psrcInfo description; Added trace ser command; Added reset message codes; Other minor changes
7	04/22/2015	Minor corrections
8	06/17/2015	Modified mxtal command to support H-Grade parts
9	12/03/2015	Added euCompliantMode and hartCompliantMode parameters
10	11/07/2016	Clarified radiotest tx command; Added mshow rstat command; Added joinshed to mset and mget commands

2 Introduction

This guide describes the commands used to communicate with the SmartMesh WirelessHART mote through its command line interface (CLI). The CLI is available by connecting a serial terminal program to the mote's CLI port. The CLI is intended for human interaction with a manager, e.g. during development, or for interactive troubleshooting. Most commands are atomic - a command and its arguments are typed into the CLI, and a response is returned. For example, the `help` command returns a list of possible commands. Traces are not atomic - once started, they generate output asynchronously until cancelled.

For a machine-to-machine communications (e.g. a sensor application talking to the mote), the SmartMesh WirelessHART Mote API Guide is used. See the API guide for details on that interface.

Only WirelessHART motes based on the LTC5800 have a CLI port - older motes do not have a CLI.

2.1 CLI Access

There are two dedicated serial ports on the SmartMesh WirelessHART mote: one is for API communication with an external application, and the other is dedicated to this CLI.

You can access the CLI from any serial terminal program (such as HyperTerminal):

- Serial 0 — If connecting to an evaluation board integrated with an FTDI serial-to-usb interface, the CLI will be found on the **3rd COM** port mapped onto your system.

The default serial port settings are as follows:

- Bits per second: 9600
- Data bits: 8
- Parity: None
- Stop bits: 1
- Flow control: None

3 Commands

3.1 get

Description

Get application parameters.

Syntax

```
get <parameter>
```

Parameters

Parameter	Description
mode	Returns the current mode (*master* or *slave*)

Example

```
> get mode
master
```

3.2 help

Description

Show help. Entering this command without parameters displays the list of all available commands. Help on a specific command may be obtained by entering that command as an argument.

Syntax

```
help [command]
```

Parameters

Parameter	Description
command	Any of the CLI commands

Example

```
> help set
Usage:
set mode [master|slave]
```

3.3 info

Description

Displays various information about the application layer, such as app name, the state, version, etc.

Syntax

```
info
```

Parameters

Parameter	Description

Example

```
> info
HART Mote 1.0.0-104
Join state:n/a
Bandwidth Allocated:0
Serial mode:Mode 1
Serial Baud Rate:115200
```

3.4 loc

Description

Send a local command to the net layer. This command is intended for internal mote development, evaluation, and advanced use as directed by an application note.

Syntax

```
loc <payload>
```

Parameters

Parameter	Description
payload	Binary string up to 90 bytes in length

Example

```
> loc 0102030405
```

3.5 mclearnv

Description

Clears nonvolatile storage of application parameters.

Syntax

```
mclearnv
```

Parameters

Parameter	Description

Example

```
> mclearnv
```

3.6 mfs

Description

File system commands. These are intended for debugging.

> The zeroize command will render the mote inoperable. It must be re-programmed via SPI or JTAG in order to be useable.

Syntax

mfs <cmd> {-f|-p} [<param>...]

Parameters

Parameter	Description
cmd	One of: show - show a list of files (-f) or partitions (-p) fcs - calculate CRC for a filename (-f <filename>) or partition (-p <parId> <offset> <length>) zeroize <password> - zeroize device keys per FIPS-140 requirements. password is 57005 (0xDEAD).

Example

> mfs show -p
ID Size Address Page
1 32768 0x000b7800 2048 exec
2 258048 0x00041000 2048 exec
4 227328 0x00080000 2048
6 2048 0x000bf800 2048

3.7 mget

Description

Used to get MAC layer parameters.

Syntax

```
mget <parameter>
```

Parameters

Parameter	Description
netid	Network ID
macaddr	MAC address (EUI-64), e.g. 01-23-45-67-89-AB-CD-EF
joincntr	Current value of join counter
txpwr	Tx power (dBm)
jkey	Returns an error as join key is write only
psrcInfo	Power source info. Concatenation of 4 subfields (entered as hexidecimal in the form aabbbbccccccddddddddd), where aa = Power source (00=line, 01= battery, 02=rechargeable) bbbb = Max discharge current (µA, little-endian. e.g 1000 is entered as e803) cccccccc = Max time at discharge current (s) ddddddddd = Time to recover after discharged (s)
ttl	Time-to-live (hops)
antgain	Antenna Gain (dBi, defaults to 2 dBi)
otaplock	OTAP lockout status. If 1, device cannot be OTAP'd
joinmode	HART join mode - see HART Spec 155 and 183 for a description of join mode
joinshed	Join shed time - see HART Spec 155 for a description of join shed time

compliantMode (1.1.x) or hartCompliantMode (>= 1.2.x)	Sets the join delay, number of join parents, and health report counter mode to those defined in the HCF specifications. 0=off (default), 1 = on. Note: This parameter is available in devices running mote software >= 1.1.0
euCompliantMode	Constrains mote duty cycle to power-appropriate limits imposed by EN 300 328. 0=off (default), 1 = on.

Example

```
> mget netid
netid=1229
```

3.8 mgeti

Description

Get internal MAC state machine parameters. These are intended for internal mote development, evaluation, and advanced use as directed by an application note.

Syntax

```
mgeti <param>
```

Parameters

Parameter	Description
pftimer	Path fail timer (in milliseconds)
advtimer	Interval (in seconds) between advertisments
srmaxretr	Maximum number of retries on source routes
joindc	Join Duty Cycle
traceflgs	Shows what traces are enabled
nwl	Network White List - See mseti for complete description. Note: This parameter is available in devices running mote software >= 1.1.0

Example

```
> mgeti pftimer
pftimer=60
```

3.9 minfo

Description

This command will return information about the mote, namely the code version, current join state, MAC address, Mote ID, Network ID, bootloader version, UTC time, and reset status.

Syntax

```
minfo
```

Parameters

Parameter	Description

Example

```
> minfo
HART stack ver 1.0.0 #1
state:       Init
mac:         00:17:0d:00:00:30:00:70
moteid:      0
netid:       1229
blSwVer:     9
UTC time:    2363:960000
reset st:    0x600
```

3.10 mset

Description

Used to set MAC layer parameters. This change is persistent.

Syntax

```
mset <param> <value>
```

Parameters

Parameter	Description
netid	Network ID. As of version 1.1.1, a network ID of 0xFFFF can be used to indicate that the mote should join the first network heard.
macaddr	MAC address (EUI-64), e.g. 01-23-45-67-89-AB-CD-EF
joincntr	Join counter
txpwr	Tx power (dBm)
jkey	Join key - must match that on manager
psrcInfo	Power source info. Concatenation of 4 subfields (entered as hexadecimal in the form aa:bb:bb:cc:cc:cc:cc:dd:dd:dd:dd), where aa = Power source (00=line, 01= battery, 02=rechargeable) bbbb = Max discharge current (µA, little-endian. e.g 1000 µA is entered as e803) cccccccc = Max time at discharge current (s) dddddddd = Time to recover after discharged (s)
ttl	Time-to-live (hops)
antgain	Antenna Gain (dBi) - defaults to 2 dBi
otaplock	OTAP lockout status. If 1, device cannot be OTAP'd
joinmode	HART join mode - see HART Spec 155 command 771 and HART Spec 183 (tables) for a description of join mode. Note that unlike with 771, the mote will not reset when setting joinmode.
joinshed	Active search (join) shed time - see HART Spec 155 command 771 for a description of active search shed time

compliantMode (1.1.x) or hartCompliantMode (>= 1.2.x)	Sets the join delay, number of join parents, and health report counter mode to those defined in the HCF specifications. 0=off (default), 1 = on. Note: This parameter is available in devices running mote software >= 1.1.0
euCompliantMode	Constrains mote duty cycle to power-appropriate limits imposed by EN 300 328. 0=off (default), 1 = on.

Example

```
> mset macaddr 01-23-45-67-89-AB-CD-EF
> mset psrcInfo 01:e8:03:ff:ff:ff:ff:00:00:00:00
```

3.11 mseti

Description

Set internal MAC state machine parameters. These are intended for internal mote development, evaluation, and advanced use as directed by an application note. This change is persistent and used the next time the mote reboots.

Syntax

```
mseti <param> <value>
```

Parameters

Parameter	Description
pftimer	Path fail timer (in milliseconds) - normally overwritten by manager at join
advtimer	Interval (in seconds) between advertisments - normally overwritten by manager at join.
srmaxretr	Maximum number of retries on source routes - defaults to 5
joindc	Join Duty Cycle
nwl	Network White List - sets neighbor that mote can join through (1st in the list), and neighbors that mote can discover (the rest in the list). Up to total of 8 neighbors, IDs supported are 1 byte only. Note: This parameter is available in devices running mote software >= 1.1.0

Example

```
> mseti nwl 2 3 4 5
```

3.12 mstacks

Description

Shows information about low level resources. Intended for debugging.

Syntax

```
mshow <object>
```

Parameters

Parameter	Description
object	One of: stacks - list of tasks and amount of RAM used (in 32-bit words) tasktime - amount of time processor is idle or executing tasks rstat - extended radiotest stats including average RSSI and LQI. Available in mote 1.2.0 or later.

Example

```
> mshow stacks
Task ?          used   50 of  256 (19%)
Task ?          used   87 of  192 (45%)
Task drvMon     used   80 of  128 (62%)
Task macCtrl    used  110 of  256 (42%)
Task net        used   41 of  384 (10%)
Task loc        used   88 of  384 (22%)
Task locNotifTask used   75 of  260 (28%)
Task moteMon    used  146 of  384 (38%)
Task clitask    used  134 of  180 (74%)
Task uC/OS-II Idle used   19 of   64 (29%)
flash err = 0
flash err cnt = 0
```

3.13 mtrace

Description

Turn MAC layer traces on or off. If called with no arguments, returns current state of all mtraces.

Syntax

```
mtrace [<parameter> {on | off}]
```

Parameters

Parameter	Description
mac	MAC layer transmit and receive events
io	Description of the commands in the packet
otap	Progression/status of the over the air programming
all	All trace elements

Example

```
> mtrace mac on
RAM mtrace settings = 16
```

3.14 mtracesv

Description

Save active traces to NV memory.

Syntax

```
mtracesv
```

Parameters

Parameter	Description

Example

```
> mtracesv
RAM mtrace settings of 16 saved to NVRAM
```

3.15 mxtal

Description

This command is used to determine the optimal trim value to center the 20MHz crystal oscillator frequency given a particular PCB layout and crystal combination. It is used to measure the 20 MHz crystal, after which the user must enter trim values into the device's fuse table for access by software. See the Board Specific Configuration Guide for fuse table details.

An additional optional temperature grade argument is available in mote >= 1.1.3. The command will return an error if the part is tested using incorrect temperature grade parameters.

Syntax

```
mxtal [trim|meas] [<i>|<h>]
```

Parameters

Parameter	Description
trim	Trims the adjustable load capacitance for the 20MHz crystal to match the frequency reference on the DC9010 programming board. Outputs the post-trim ppm error and the optimal value of the load-capacitance setting. The trimmed value of the load capacitance is not stored in the mote application, rather in a custom fuse table; the function output should be used to determine the the proper value of the load-capacitance setting for the BSP fuse table parameter. This function requires the mote be connected to the DC9010 programming board. It could take up to 30 sec for command to execute.
meas	Outputs the ppm error of the 20MHz reference with value loaded from the fuse table . This function requires the mote be connected to the DC9010 programming board. It could take up to 30 sec for command to execute.
i | h	Temperature grade, one of i=industrial or h=high temperature - See device datasheet for details. Defaults to i (industrial) if omitted.

Example

On an i-grade part:

```
> mxtal meas
Fuse Table pullVal used for measurement=95

> mxtal trim i
The optimal pullVal for this board is 90, which yields 0/16 PPM error
```

3.16 radiotest

3.16.1 radiotest on/off

Description

Enable or disable radiotest mode on the device. Radiotest functionality can be used to exercise the radio for certification and testing purposes. This command takes effect after reboot and the selected mode persists until changed, i.e. if ON, it will remain on even after reset or power cycle until the mode is set to OFF and the device is rebooted.

Syntax

```
radiotest <mode>
```

Parameters

Parameter	Description
mode	on - put device into radiotest mode after reboot off - put device into normal master mode after reboot

Example

Put device into radiotest mode:

```
radiotest on
```

Return device to normal operational mode:

```
radiotest off
```

3.16.2 radiotest tx

Description

The `radiotest tx` command allows the user to initiate a radio transmission test. This command may only be issued in radiotest mode. Three types of transmission tests are supported:

- pk - Packet Transmission
- cm - Continuous Modulation
- cw - Continuous Wave (unmodulated signal)
- pkcca - Packet transmission with clear channel assessment (CCA) enabled (Available in IP Manager >= 1.3.0 and IP mote >= 1.4.0)

In a packet transmission test, the device generates a *repeatCnt* number of packet sequences. Each sequence consists of up to 10 packets with configurable sizes and delays. Each packet consists of a payload of up to 125 bytes, and a 2-byte 802.15.4 CRC at the end. Byte 0 contains sender's stationId. Bytes 1 and 2 contain the packet number (in big-endian format) that increments with every packet transmitted. Bytes 3..N contain a counter (from 0..N-3) that increments with every byte inside payload. Transmissions occur on the set of channels defined by *chanMask*, selected in pseudo-random order.

In a continuous modulation test, the device generates continuous pseudo-random modulated signal, centered at the specified single channel. The test is stopped by resetting the device.

In a continuous wave test, the device generates an unmodulated tone, centered at the specified single channel. The test tone is stopped by resetting the device.

In a packet transmission with CCA test, the device is configured identically to that in the packet transmission test, however the device does a clear channel assessment before each transmission and aborts that packet if the channel is busy.

Channel numbering is 0-15, corresponding to IEEE 2.4 GHz channels 11-26.

stationId is available in SmartMesh IP Mote >= 1.4, SmartMesh IP Manager >= 1.3.0, SmartMesh WirelessHART mote >= 1.1.2

Syntax

```
radiotest tx <testType> <chanMask> <power> [<stationId> <repeatCnt> {<pkLen><delay>...}]
```

Parameters

Parameter	Description
testType	Type of transmission test to initiate: 'pk' = packets, 'cm' = continuous modulation, 'cw' - continuous wave, "pkcca" = packets with CCA.
chanMask	Hexadecimal bitmask of channels (0–15) for the test. Bit 0 corresponds to channel 0. For continuous wave and continuous modulation tests, only one channel should be enabled.
power	Transmit power, in dB. Valid values are 0 and 8.
stationId	Unique (0-255) station id of the sender. Must match station id value of the receiver.
repeatCnt	Number of times to repeat the packet sequence (0=do not stop). Applies only to packet transmission tests.
pkLen	Length of packet (2-125 bytes)
delay	Delay after transmission (0-65535 microseconds)

Example

Initiate packet test on channels 0,1 (chMap=0x03), with output tx power of 0 dBm, station id = 26
Repeat the sequence 5 times: 50-byte packet, 20ms delay, 30-byte packet, 20msec delay

```
radiotest tx pk 0x3 0 26 5 50 20000 30 20000
```

Start transmission with continuous modulation on channel 0 with output tx power of 8 dB

```
radiotest tx cm 0x1 8
```

Start transmission with continuous wave on channel 1 with output tx power of 8 dB

```
radiotest tx cw 0x2 8
```

3.16.3 radiotest rx

Description

The `radiotest rx` command puts the radio into receive mode where statistics on packet reception are collected. The nonzero station id specified must match station id of the sender, which is necessary to isolate traffic of multiple tests running in the same radio space. Statistics may be viewed with the `radiotest stat` command.

⚠ Channel numbering is 0-15, corresponding to IEEE 2.4 GHz channels 11-26.

⚠ stationId is available in SmartMesh IP Mote >= 1.4, SmartMesh IP Manager >= 1.3.0, SmartMesh WirelessHART mote >= 1.1.2

Syntax

```
radiotest rx <chanMask> <time> <stationId>
```

Parameters

Parameter	Description
chanMask	Hexadecimal bitmask of channels (0–15) for the test. Bit 0 corresponds to channel 0. Only a single channel may be specified for this command.
time	Duration of receive test, in seconds. 0=do not stop
stationId	Unique (1-255) id of the receiver. Must match sender's station id. Station id 0 may be used to accept packets from any sender.

Example

Put device into receive mode for 60 seconds on channel 2, use station id 26:

```
radiotest rx 0x4 60 26
```

3.16.4 radiotest stat

Description

The `radiotest stat` command displays packet reception statistics collected during the previously run `radiotest rx` command. This command may only be used when the device is in radiotest mode.

Syntax

```
radiotest stat
```

Parameters

Parameter	Description

Example

```
>radiotest stat
Radio Test Statistics
  OkCnt   : 0
  FailCnt : 0
```

3.17 reset

Description

Reset the mote.

Syntax

```
reset
```

Parameters

Parameter	Description

Example

```
> reset
HART Mote 1.0.0-104
```

3.18 set

Description

Set application parameters. This change is persistent.

Syntax

```
set <parameter> <value>
```

Parameters

Parameter	Description
mode	master - the application will terminate local commands slave - the local commands will be forwarded to the serial port

Example

```
> set mode slave
```

3.19 trace

Description

Turn application layer traces on or off. If called with no arguments, returns current state of all traces. If called with the argument "save" it stores current settings to non-volatile memory.

Syntax

```
trace [save | {<module>|all on|off}]
```

Parameters

Parameter	Description
module	One of: loc - local (net layer) commands oap - application commands ser - serial commands all - reserved rc - reserved sm - reserved

Example

```
> trace loc on
```

4 Error Messages

Mote software is organized into various OSI-model layers, e.g. the Medium Access Control (MAC) layer is responsible for packet delivery between neighbors, while the Network (NET) layer handles end-to-end delivery. When a stack layer encounters an error, it will be printed on the CLI, starting with an OS timestamp in ms. For example:

```
39557 : MAC retry drop
```

The following tables explain the meaning of the various error messages.

MAC layer	Meaning
PF: n= t= lh= d=	Path failure to neighbor 'n' at time 't', last communication time 'lh', 'd' is difference between 't' and 'lh'
MAC retry drop	Packet is dropped because number of source-route retries is exceeded
MAC pdu tout	Packet is dropped because of PDU timeout
MAC no route	Packet is dropped because the graph to send it on was not found
Disconnecting	MAC started disconnecting process
RX ADV SYNC failed	Failed synchronization time bounds check when processing advertisement
listen chan =	Channel switched in promiscuous mode (searching for network)

Local (API) layer	Meaning
Event NACKed	Local interface received DN_ERR_NO_RESOURCES to sent event
Rx Notif NACKed	Local interface received DN_ERR_NO_RESOURCES to sent packet received notification
Time Notif NACKed	Local interface received DN_ERR_NO_RESOURCES to sent time notification

Filesystem layer	Meaning
'Filename' could not be created	Could not open a file in file system
Error while deleting 'filename'	Could not delete a file in file system

Network layer	Meaning
Join failed	Failed to join after maximum number of join retries
Disconnected	Received *disconnected* event notification from MAC

The mote will print a bitmap indicating the reason for the last reset

Reset Status	Meaning
0x100	Watchdog
0x200	External reset pin asserted
0x400	Power-on
0x800	Brownout
0x40000	FLASH_P_EN was asserted at boot
0x40000000	CPU Lockup
0x80000000	Sysreset

Trademarks

Eterna, Mote-on-Chip, and SmartMesh IP, are trademarks of Dust Networks, Inc. The Dust Networks logo, Dust, Dust Networks, and SmartMesh are registered trademarks of Dust Networks, Inc. LT, LTC, LTM and  are registered trademarks of Linear Technology Corp. All third-party brand and product names are the trademarks of their respective owners and are used solely for informational purposes.

Copyright

Disclaimer

This documentation is provided “as is” without warranty of any kind, either expressed or implied, including but not limited to, the implied warranties of merchantability or fitness for a particular purpose.

This documentation might include technical inaccuracies or other errors. Corrections and improvements might be incorporated in new versions of the documentation.

Linear Technology does not assume any liability arising out of the application or use of any products or services and specifically disclaims any and all liability, including without limitation consequential or incidental damages.

Linear Technology products are not designed for use in life support appliances, devices, or other systems where malfunction can reasonably be expected to result in significant personal injury to the user, or as a critical component in any life support device or system whose failure to perform can be reasonably expected to cause the failure of the life support device or system, or to affect its safety or effectiveness. Linear Technology customers using or selling these products for use in such applications do so at their own risk and agree to fully indemnify and hold Linear Technology and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that Linear Technology was negligent regarding the design or manufacture of its products.

Linear Technology reserves the right to make corrections, modifications, enhancements, improvements, and other changes to its products or services at any time and to discontinue any product or service without notice. Customers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All products are sold subject to Dust Network's terms and conditions of sale supplied at the time of order acknowledgment or sale.