SmartMesh WirelessHART Easy Start Guide

Table of Contents

1 Introduction 3
1.1 Revision History 3
2 Setup 4
3 Basic Steps 5
3.1 Step 1: Connect to the Manager CLI 5
3.2 Step 2: Form a Wireless Mesh Network 6
3.3 Step 3: Establish PC Connection to a Mote 6
3.4 Step 4: Connect to Mote CLI 7
3.5 Step 5: Install SDK Software 8
3.6 Step 6: Join a Mote to the Manager with the API 8
4 Additional Tools 10
5 Steps in a Design 11
6 Troubleshooting 12
6.1 References 14

1 Introduction

The purpose of this document is to get the user a quick-and-easy, positive out-of-the-box experience with the SmartMesh WirelessHART Starter Kit and Software Development Kit (SDK). More details are discussed in the SmartMesh WirelessHART Tools Guide. A number of software components are available for demonstrating interaction with your network. In this document, we will use the components highlighted in gray: FTDI drivers for converting USB connections to virtual COM ports; a terminal application; and the SmartMesh SDK, which allows the user to interact graphically with mote and manager APIs.

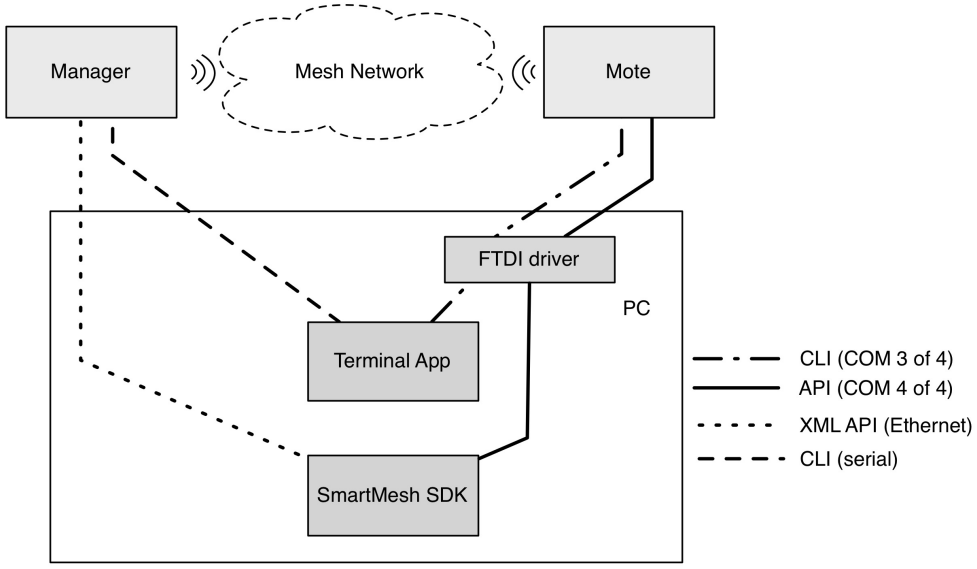


Figure 1: SmartMesh Software Components

1.1 Revision History

Revision	Date	Description
1	03/18/2013	Initial Release
2	09/30/2013	Added DC9022

2 Setup

Figure 1 below illustrates the required SDK setup. The kit includes a SmartMesh WirelessHART Manager (LTP5903CEN-WHR), five Eterna Motes, and an interface board (DC9006A). There are different manager/mote combinations available, as shown in Table 1.

Kit	Mote	Manager	Notes
DC9007A	DC9003A-C	LTP5903CEN-WHR	Eval/Dev Kit with packaged manager
DC9022A	DC9018A-C	LTP5903CEN-WHR	RF Certified Eval/Dev kit with packaged manager

All motes in kits ship with chip antennas. RF Certified devices are available for order individually with MMCX connectors:

- Mote - DC9018B-C (MMCX connector)

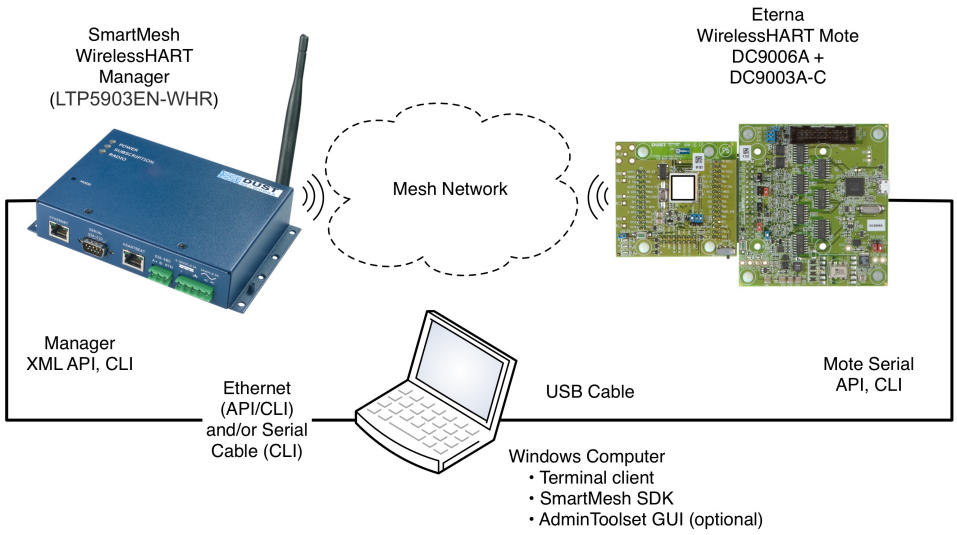


Figure 1: SmartMeshSDK Setup

3 Basic Steps

There are six steps to get the SDK running:

1. Connect to the Manager CLI
2. Form a wireless mesh network - just power everything on, and a network will form
3. Establish a connection between your PC and a Mote
4. Connect to the Mote CLI
5. Install SDK Software
6. Connect to Manager API and Mote API

Once you've done these 6 steps you'll perform one exercise - you'll use the Mote API to make a Mote join the Manager.

3.1 Step 1: Connect to the Manager CLI

Even though much of the effort associated with this guide involves using the PC, it is important to point out that your PC has nothing to do with forming a wireless mesh network. It is only there to allow you to visualize the network and interact with it while it is forming and once it has formed.

The manager command line interface gives you a text-based way to interact with a manager. You will need to connect a DB-9 serial cable between the LTP5903EN-WHR port labeled "Serial 2" and your computer. If you do not have a DB-9 serial port on your computer, you will need to use a USB-serial adapter (e.g. a Keyspan 19HS) to enable CLI.

Interacting with the CLI requires no software installation on the PC if you are running Windows XP, as it comes with the Hyperterminal client. If you are running Windows 7, you will need to download a terminal client (such as PuTTY). A list of other terminal clients can be found in the SmartMesh WirelessHART Tools Guide. The port settings are:

115200 baud, 8 data bits, No parity, 1 stop bit, no flow control

At the prompt within the terminal program, access the Linux login prompt by entering the following username and password:

Username: `dust`
Password: `dust`

This will bring you to the Linux prompt. At the Linux prompt ($):

1. Enter: `nwconsole`
2. Enter the manager CLI user name and password. The default user name and password is:
   Username: `admin`
   Password: `admin`

3.2 Step 2: Form a Wireless Mesh Network

The network does not need human interaction to form - motes and manager are pre-configured with security and network information and will form a network by themselves once powered up. To illustrate this, power on the Manager (LTP5903CEN-WHR). Note the Power LED. That indicates that the Manager is on and is attempting to form a network by sending out advertisements.

Type the following command into the manager CLI:

```
trace motest on
```

This will cause the manager to print a message every time a mote makes a state transition.

Power on one Mote (DC9003A-C). Wait until you see the mote transition to **Operational** (Oper). This will take 1-2 minutes.

Power on all the remaining Motes, and within a few minutes you will have a complete wireless mesh network running. The Manager CLI is a useful interface for diagnostics and debugging - it is worthwhile to familiarize yourself with it. See the SmartMesh WirelessHART Manager CLI Guide for details on the various commands that are available. The rest of the steps we go through will give us additional ways to interact with the mesh network we just formed.

3.3 Step 3: Establish PC Connection to a Mote

You will now connect the PC to the Mote through your USB port. First connect the Mote to the DC9006 interface board using the side connector, if not already done. Next, plug the USB cable into one of your PC USB ports. Note which USB port you are using. You should consider this your Mote port for all future use with this PC.

Four virtual COM ports should automatically be added. You may have to install FTDI drivers if this doesn't happen automatically. Refer to the SmartMesh WirelessHART Tools Guide for detailed instructions.

Note the number of these four COM ports, they will be called something like "COM11", "COM12", "COM13" and "COM14". You will be using the third of the four for CLI and the fourth (highest chronological number) for the API connection. In this case, COM13 and COM14, correspondingly.



It has been observed in some installations under Windows 7 that the serial ports do not enumerate in order, and the CLI and API ports may not be the 3rd and 4th ports, respectively. If this occurs, you will need to test each port using APIExplorer (in the SmartMesh SDK) to find the API port, and use a terminal program to find the CLI port.

3.4 Step 4: Connect to Mote CLI

Connecting to the Mote CLI is very similar to connecting to the Manager CLI above. Connect to the third of the four COM ports that were added when you plugged in the Mote. The COM port settings are 9600 baud, 8 data bits, No parity, 1 stop bit, no flow control

Hit enter a few times to get a ">" prompt, then type the `minfo` command to get info from the Mote:

```
> minfo
HART stack ver 1.0.1 #1
state:       Oper
mac:         00:17:0d:00:00:38:0c:ae
moteid:      28
netid:       303
blSwVer:     10
ldrSwVer:    1.0.3.12
UTC time:    1355189040:1355189040
reset st:    0x100
```

Now we will "turn on" this Mote's API port. Motes in the kit were shipped to you in **master** mode. In **master** mode, the Motes join on their own, and the API is turned off. We will switch this Mote to **slave** mode. This mode turns on the API and the Mote will only join when told to do so by the sensor application. In our case today, the PC plays the role of the sensor application.

You can get the current mode of the Mote with the command:

```
> get mode
```

To change the Mote to **slave** mode, type the two commands:

```
> set mode slave
> reset
```



Reset is required for the `set mode slave` command to take affect.

Verify that you have set the Mote to **slave** by executing the `get mode` command again.

3.5 Step 5: Install SDK Software

While the CLI is designed for direct human interactions through text input, the API is a device interface for interacting with other processors. The SDK contains applications that perform various useful functions exercising the APIs through a GUI interface. The SDK is based on Python, but you do not need to install Python to use the pre-compiled applications.

Download the latest rev of SmartMeshSDK zip file.

Unzip the file and a folder by the same name will be created with 4 sub-folders: `api`, `doc`, `src`, and `win`. Each of these has sub-folders. The important one is `win`, where executable versions of the utilities are stored. The SmartMeshSDK folder can be moved to any convenient location on your computer. NO other installation is required.

3.6 Step 6: Join a Mote to the Manager with the API

In previous steps we connected a Manager and a Mote to the computer. You should have have a terminal window connected to the CLI of the Manager and the Mote. We will now connect simultaneously to the API of the mote with the application APIExplorer.exe.



You can stay connected to the CLI terminal window while you connect to the API in a different application. This is possible because the CLI uses the third COM port for each device and the API uses the fourth COM port.

We now connect to the Mote API and make it join the Manager.

In the SmartMeshSDK directory, double click on the `win/APIExplorer.exe` application. This opens the APIExplorer window. Tell the application you want to connect to a SmartMesh WirelessHART Mote by selecting the following:

```
network type: SmartMesh WirelessHART
device type: mote
```

Click the **load** button – this loads the API for the IP Mote. In the **connection** frame, enter the following:

```
port name: your SmartMesh WirelessHART Mote's API COM port (the fourth added)
```

Recall that this is highest COM port number (the fourth port) you got in STEP 2. Click **connect**. The fields turn green indicating the connection is successful.



This connection to the Mote won't work if the Mote is in **master** mode. APIExplorer will also fail to open the port if another application is trying to use the port.

To join the Mote to the Manager, go through these four steps:

1. Issue a *getNVParameter.networkId* command to verify that your SmartMesh WirelessHART Mote is configured with the correct network ID (1229 by default)
2. Issue a *getParameter.moteStatus* command to verify that your SmartMesh WirelessHART Mote is in the Idle state
3. Issue a *join* command to tell the Mote to search and join the network
4. Repeat the *getParameter.moteStatus* command a few times over the next 30 seconds or so and you should see the Mote states proceed through to **Operational .** You should also see notifications as the mote changes state.

The Mote is now joined to the Manager.

We will now *ping* one of the Motes in the network using a Manager CLI command, e.g. here to mote 2.

```
> ping 2

17:38:25] Ping mote 28: reply #1: 3.398s 1 hops [26.0C 3.645V]
[17:38:25] Ping mote 28: sent 1, rcvd 1, 0% lost. Ave.roundtrip: 3.398s hops: 1
```

The Mote will respond with the round trip delay time, temperature and voltage as described in the SmartMesh WirelessHART Manager CLI Guide.

4 Additional Tools

Once you have demonstrated basic network formation and the ability to interact with a Mote and Manager, you are ready to install the additional tools discussed in the SmartMesh WirelessHART Tools Guide. See the SmartMesh WirelessHART Tools Guide for installation instructions. These additional tools provide additional visualization and Manager configuration options (Admin Toolset), support testing of APIs, and to generally aid in application development.

5 Steps in a Design

With the starter kit, hardware design and software design may be decoupled.

For Software design:

- The SmartMesh WirelessHART User's Guide defines basic network terms and concepts, and discusses the use of APIs at a high level
- At a minimum, a mote application needs to:
    - Configure any parameters needed prior to join (such as *joindutycyle*)
    - Use the *join* API to cause a mote to being searching for a network
    - Monitor the mote state to see when it is ready to accept data
    - Request services in order to publish data
    - The SmartMesh WirelessHART Mote API guide covers other commands to configure the mote
    - The SmartMesh WirelessHART Mote CLI guide covers using the human interface to observe mote activity
- At a minimum, a host application connected to the manager needs to:
    - Configure any parameters needed prior to join (such as *networkID*)
    - Subscribe to notifications to observe mote status and collect data
    - The SmartMesh WirelessHART Manager API guide covers other commands to configure the manager, e.g. configure security (use of ACL), or collect detailed statistics from Health Report notifications
    - The SmartMesh WirelessHART Manager CLI guide covers using the human interface to observe manager activity (including traces of mote state or data).
- Advanced software topics covered in the SmartMesh WirelessHART User's Guide, SmartMesh WirelessHART Tools guide, and SmartMesh WirelessHART Application Notes include Over-the-Air-Programming, using the TestRadio API commands for top-level assembly testing, among others

For Hardware design:

- Select a hardware platform - modularly certified or chip level?
- The hardware integration application notes and integration guides cover the important considerations for robust hardware development

6 Troubleshooting

1. When connected to a DC9006 board and a computer, the Mote may appear to be operating in spite of the power switch being off. The 4 COM ports will appear but you may not be able to communicate with the mote reliably. Make sure that the power switch on all boards is set to on to ensure proper operation.

2. If you are consistent with which USB port you plug the Mote into, the COM port assignments would should remain consistent. Please mark the physical USB ports on your machine for Mote usage and then do not change its usage.

3. Make sure that the baud rate is set to 9,600 for mote CLI and 115,200 for mote/manager API and manager CLI. As an example , see Figure 2:

- COM 7-10 are for the mote: COM10 is for API at baud rate of 115,200 and COM9 is for CLI at baud rate of 9,600

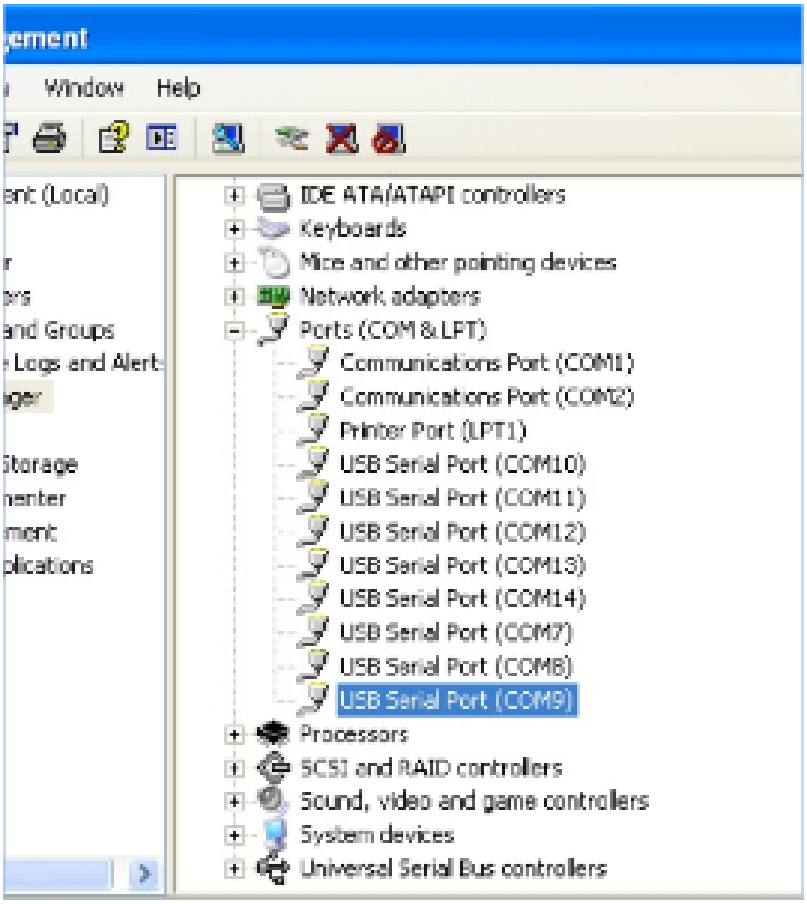


Figure 2: Device Manager Example

4. The power "slide" switch is located in corner of the mote (DC9003A-C) board. Make sure to check that the power to Manager is ON. The 4 COM ports may appear even if the mote is powered OFF, but you will not be able to communicate with the Mote.
5. To enable and use the API/interface of the Mote board (DC9003A-C), it must be programmed set to be in **slave** mode, using the CLI interface using the `set mode` command followed by a `reset` command. Once set in **slave** mode, you have to manually execute the *join* command on the Mote API interface, using APIExplorer in order to join the network. This setting is non-volatile.

6. By default, a Mote joins automatically when it boots if it is in **master** mode (default mode as shipped from the factory). In **slave** mode, a *join* command must be given.

6.1 References

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

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

This documentation is protected by United States and international copyright and other intellectual and industrial property laws. It is solely owned by Linear Technology and its licensors and is distributed under a restrictive license. This product, or any portion thereof, may not be used, copied, modified, reverse assembled, reverse compiled, reverse engineered, distributed, or redistributed in any form by any means without the prior written authorization of Linear Technology.

RESTRICTED RIGHTS: Use, duplication, or disclosure by the U.S. Government is subject to restrictions of FAR 52.227-14(g)(2)(6/87) and FAR 52.227-19(6/87), or DFAR 252.227-7015 (b)(6/95) and DFAR 227.7202-3(a), and any and all similar and successor legislation and regulation.

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.

Linear Technology does not warrant or represent that any license, either express or implied, is granted under any Linear Technology patent right, copyright, mask work right, or other Linear Technology intellectual property right relating to any combination, machine, or process in which Linear Technology products or services are used. Information published by Linear Technology regarding third-party products or services does not constitute a license from Linear Technology to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property of the third party, or a license from Linear Technology under the patents or other intellectual property of Linear Technology.

Dust Networks, Inc is a wholly owned subsidiary of Linear Technology Corporation.

© Linear Technology Corp. 2012-2013 All Rights Reserved.