



说明

本参考设计实现了一款用于智能仪表先进抄表基础设施 (AMI) 网络的简单射频网状网络终端节点。该网络是一种基于低功耗无线个人局域网的 IPv6 (6LoWPAN) 解决方案。该设计在单个 CC1310 SimpleLink™ 无线 MCU 中实现了该网络, 从而优化性能并尽可能减少系统成本。整个 6LoWPAN 网状网络都在基于 IEEE 802.15.4e/g 协议的 TI-15.4 堆栈上运行, 且实现了非时隙信道跳跃 (USCH) 模式, 可防止网络干扰。

资源

TIDA-010003	设计文件夹
CC1310	产品文件夹
TPD4E004	产品文件夹
TPD6E004	产品文件夹
TM4C1294NCPDT	产品文件夹
LM4040	产品文件夹
TPS796	产品文件夹
SN74AVC4T245	产品文件夹

特性

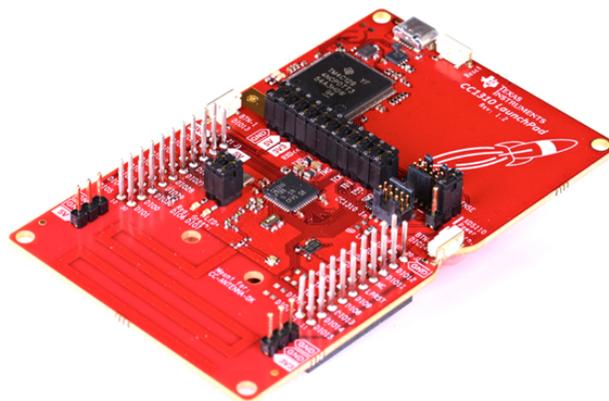
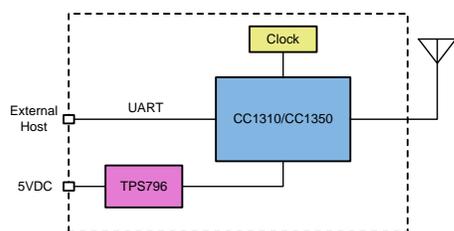
- 基于低功耗射频 (在低于 1GHz ISM 频带中运行) 的 IPv6 网络
- 实现了 6LoWPAN、RPL、IPv6、ICMPv6 和 UDP 网状网络协议
- 实现了具有基于 IEEE 802.15.4e 的跳频和 MAC 数据加密功能的 TI-15.4 堆栈
- 与 Wi-SUN FAN v1.0 相同的软件层架构
- 与 TIDA-01547、TIDA-010024 和 TIDA-010032 完全兼容的参考设计, 可提供完整的网络解决方案
- 支持适用于电池供电型终端节点的低功耗模式配置

应用

- [无线通信](#)
- [电表](#)
- [水表](#)
- [燃气表](#)



咨询我们的 E2E™ 专家



该 TI 参考设计末尾的重要声明表述了授权使用、知识产权问题和其他重要的免责声明和信息。

1 System Description

This reference design provides a simple low-power end node that supports 6LoWPAN mesh protocols. A primary design goal is to improve AMI network performance by using frequency hopping (FH) techniques that increase robustness in noisy radio frequency (RF) environments.

FH is a technique of transmitting data by switching one of many channels, where the channel is selected by a pseudo-random sequence known to both sender and receiver. This technique is known as robust versus interference and excellent in coexistence performance. As the communication systems run in sub-1 GHz ISM bands and multiple wireless systems run together, it is important that the system performance is not affected by interference or the coexisting systems. 3.2 节 shows experimental results to verify this by measuring network performance in the existence of out-of-network interference. In addition, 表 1 summarizes key system performance to show network robustness with background noise interference.

Another segment of this design is the 6LoWPAN mesh stacks, which improves network coverage and supports IPv6-based applications. The increased network coverage reduces the total system cost by reducing the number of data collectors that are typically more expensive than smart meters. The smart meters are static in the AMI networks. The mesh networking addresses the connectivity issue through multi-hop transmissions when data collector and smart meters are not reachable with each other.

This design is based upon a SimpleLink MCU of the CC13x0 wireless MCU. 图 1 shows the overall system architecture. The TI-15.4 Medium Access Control (MAC) supports FH mode. The TI SimpleLink PHY supports IEEE 802.15.4g for 50-kbps, 200-kbps frequency-shift keying (FSK) modes and 5-kbps long range mode. The sub-1 GHz RF on the CC1310 MCU can support three frequency bands: 902 MHz, 863 MHz, and 433 MHz.

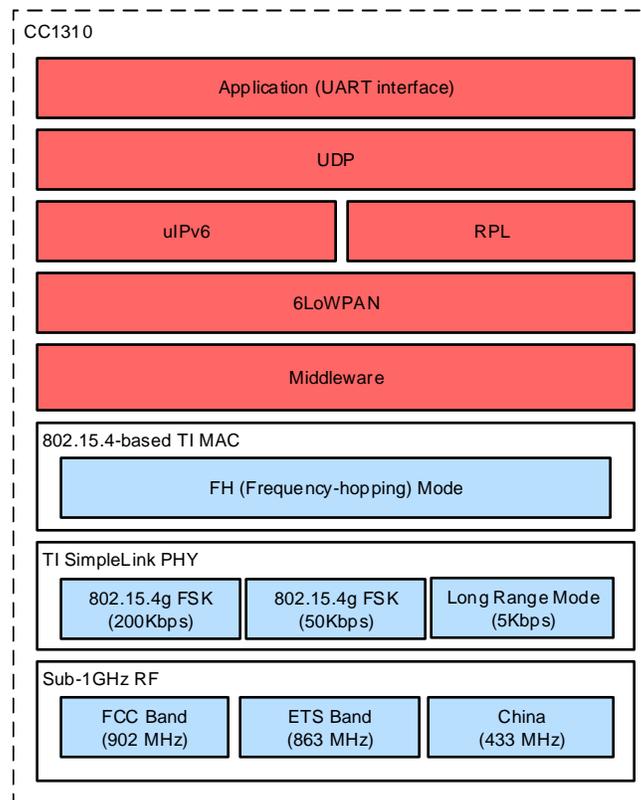


图 1. TIDA-010003 System Architecture

1.1 Key System Specifications

表 1. Key System Specifications

PARAMETER	SPECIFICATIONS	DETAILS
Maximum number of hops	<ul style="list-style-type: none"> 64 hops (in software, configurable) Tested up to 6-hop networks 	
Maximum number of route entries	<ul style="list-style-type: none"> 10 (in software, configurable) Tested up to 20-node networks 	The overall network size will depend on network topology, not limited by the maximum number of entries.
Maximum number of neighbor nodes	<ul style="list-style-type: none"> 10 (in software, configurable) Tested up to 20-node networks 	
Maximum application data size	<ul style="list-style-type: none"> 100B (in software, configurable) Tested up to 100B 	
Delivery ratio	<ul style="list-style-type: none"> 99.7% (average in 6-hop linear topology with continuous background noise) 	节 3.2.2.2
Round-trip time (RTT)	<ul style="list-style-type: none"> 0.19 second (100B over 1-hop node) 1.17 second (100B over 6-hop node) 	节 3.2.2.3
Goodput	<ul style="list-style-type: none"> 8.3 kbps (100B in 1-hop node) 	节 3.2.2.4
MAC data encryption	<ul style="list-style-type: none"> IEEE 802.15.4-based encryption supported MIC-32, MIC-64, MIC-128 ENC, ENC-MIC-32, ENC-MIC-64, and ENC-MIC-128 	节 3.1.2.2.1.3
Flash, RAM usage	<ul style="list-style-type: none"> Flash: 120KB (128KB in total) RAM: 22KB (20KB + 8KB (cache RAM) in total) 	

2 System Overview

2.1 Block Diagram

图 2 shows the system block diagram. The CC1310 (or CC1350) MCU is the 6LoWPAN mesh MCU that runs UDP applications, 6LoWPAN mesh network and the TI 15.4-Stack over sub-1 GHz RF.

The external DC/DC converter, as shown in 图 2, is needed when the external power source supplies the voltage level other than 3.3 V. In this reference design, because the evaluation modules (EVMs) are powered by USB, the TPS796 was chosen to convert 5 V to 3.3 V.

For end-equipment designs, the selection of the power supply depends on the input and output voltage level, and required current consumption. The [TI WEBENCH Power Designer](#) provides the detailed design of the power supply with the given input requirements.

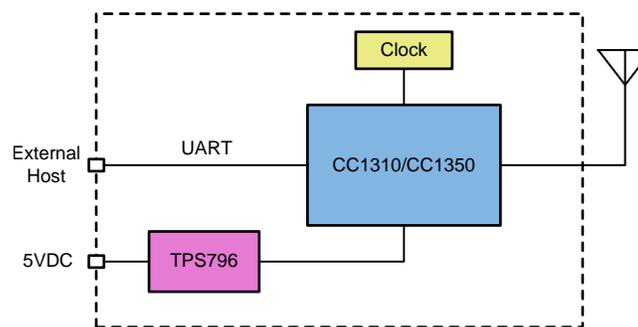


图 2. TIDA-010003 Block Diagram

2.2 Design Considerations

For this reference design, these devices perform the following:

- The CC1310 wireless MCU combines an ARM Cortex-M3 MCU with a flexible, ultra-low-power RF transceiver with excellent RX sensitivity to provide a robust link budget and execute the TI 15.4-Stack.
- The TPS796 low-power linear regulator offers high power-supply rejection ratio (PSRR), ultra-low noise, fast start-up, and excellent line and load transient responses.

2.3 Highlighted Products

2.3.1 CC1310

The CC1310 wireless MCU is a member of the Simplelink MCU platform. Its very low current consumption in both active and standby mode provides excellent lifetime when operating from batteries or super capacitors.

The CC1310 combines a flexible, very low-power RF transceiver with a powerful 48-MHz Cortex-M3 MCU in a platform supporting multiple physical layers and RF standards. A dedicated radio controller (Cortex-M0) handles low-level RF protocol commands that are stored in ROM or RAM, thus ensuring ultra-low power and flexibility. Its RF subsystem offers an excellent link budget with receiver sensitivity with -100 dBm at 50 kbps and output power up to $+15$ dBm. The CC1310 is a highly integrated, true single-chip solution incorporating a complete RF system and an on-chip DC/DC converter. The CC1310 wireless MCU is supported by the SimpleLink Software Development Kit (SDK) that offers 100% application code compatibility across the entire SimpleLink MCU portfolio and includes the integrated TI-RTOS, complete peripheral driver libraries with POSIX-compatible APIs, and encryption-enabled security features.

图 3 shows the CC1310 functional block diagram.

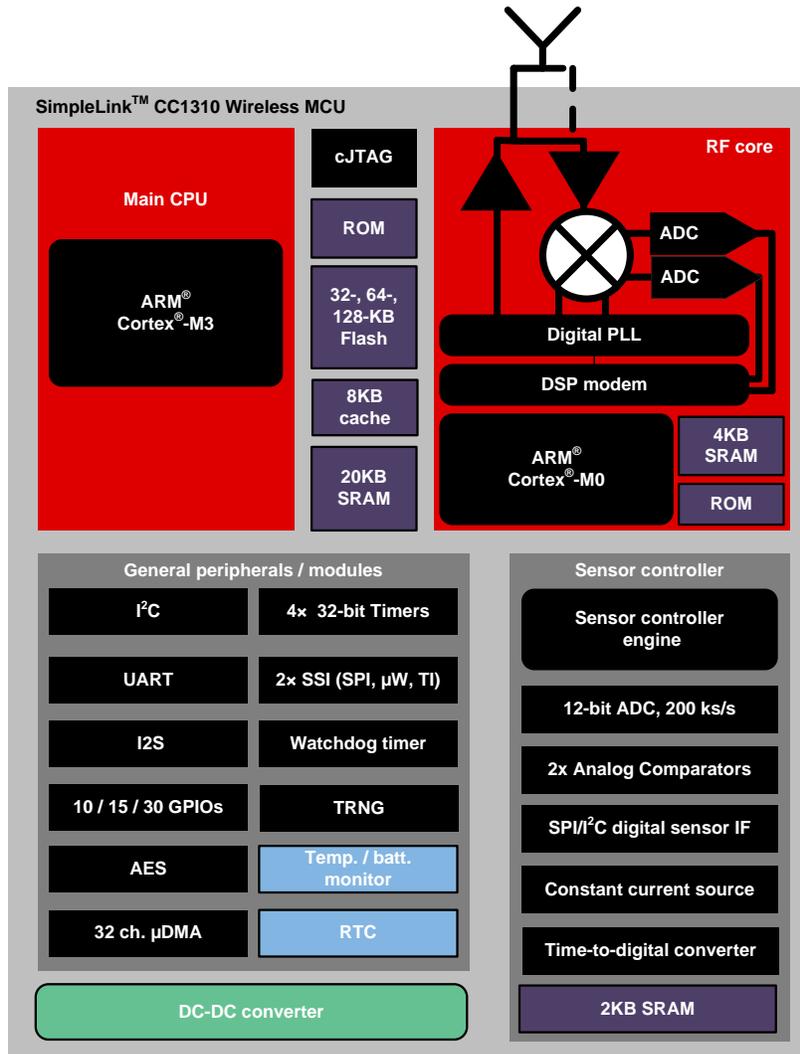


图 3. CC1310 Functional Block Diagram

2.3.2 TPS796

The TPS796 family of low-dropout (LDO) low-power linear voltage regulators feature high power-supply rejection ratio (PSRR), ultra-low noise, fast start-up, and excellent line and load transient responses in small outline, 3x3 VSON, SOT223-6, and TO-263 packages. Each device in the family is stable with a small, 1- μ F ceramic capacitor on the output. The family uses an advanced, proprietary BiCMOS fabrication process to yield extremely LDO voltages (for example, 250 mV at 1 A).

Each device achieves fast start-up times (approximately 50 μ s with a 0.001- μ F bypass capacitor) while consuming very low quiescent current (265 μ A, typical). Moreover, when the device is placed in standby mode, the supply current is reduced to less than 1 μ A. The TPS79630 exhibits approximately 40 μ V_{RMS} of output voltage noise at 3.0-V output with a 0.1- μ F bypass capacitor. Applications with analog components that are noise sensitive, such as portable RF electronics, benefit from the high PSRR, low-noise features, and fast response time.

图 4 和 图 5 显示 TPS796 的功能块图。

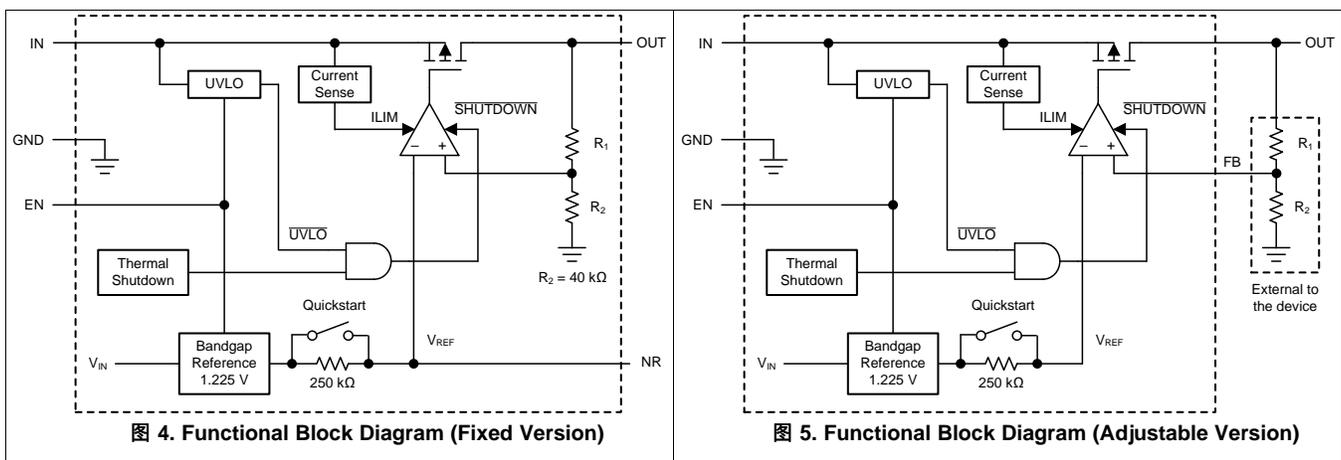


图 4. Functional Block Diagram (Fixed Version)

图 5. Functional Block Diagram (Adjustable Version)

3 Hardware, Software, Testing Requirements, and Test Results

3.1 Required Hardware and Software

3.1.1 Hardware

This reference design is built with a standard TI EVM of [LAUNCHXL-CC1310](#), as shown in [图 6](#). The CC1310 EVM can be replaced with the CC1350 EVM ([LAUNCHXL-CC1350](#)) with the same software example.

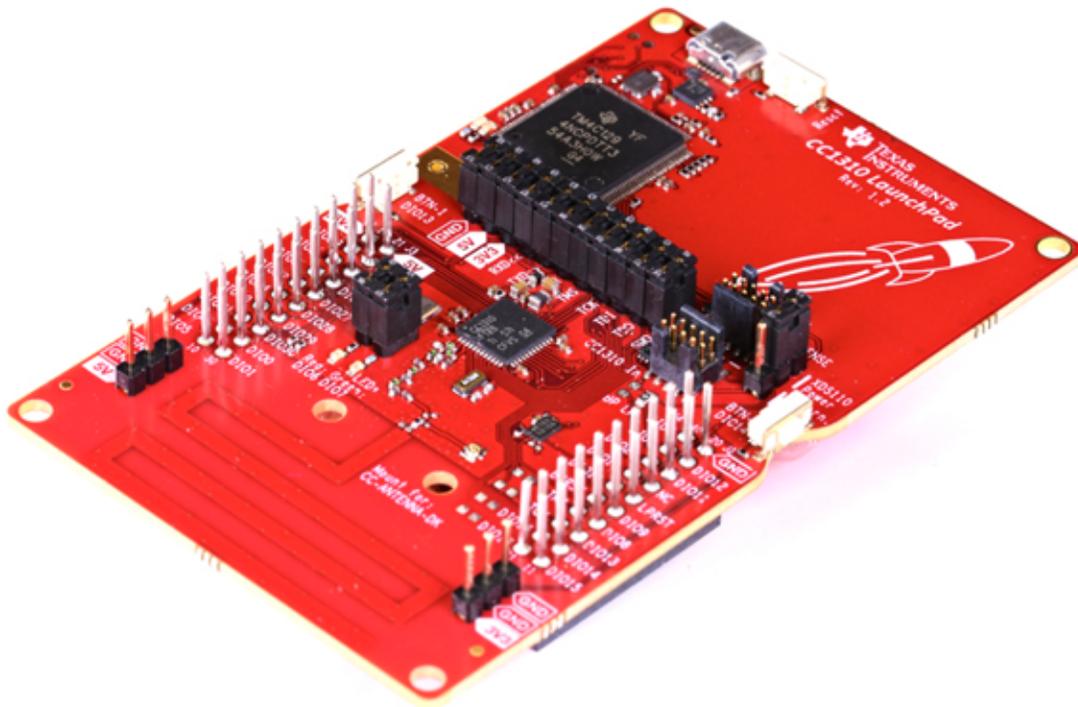


图 6. TIDA-010003 Hardware Platform

注: If a software extension with more features is considered, it is strongly recommended to choose the CC13x0 with 7x7 package, which is pin-to-pin compatible to CC13x2 with more memory option.

3.1.2 Software

3.1.2.1 Getting Started

The 6LoWPAN mesh end-node software example was implemented based on the TI-15.4 sensor example from SimpleLink CC13x0 SDK v2.10.00.36. The software example provided with the reference TI design runs on the CC13x0 MCU to support 6LoWPAN, RPL routing, IPv6, ICMPv6, UDP, and simple applications. The pre-requisite tools to build the software example are Code Composer Studio™ v8.0 or above ([CCSTUDIO](#)) and SimpleLink CC13x0 SDK v2.40 ([SIMPLELINK-CC13X0-SDK](#)).

图 7 shows the software example that provides four end-node configuration options. They are UDP poll, UDP push, UDP poll for demo with TIDA-010032 and UDP push for leaf configurations. For the UDP poll example, end nodes send UDP data only when they receive the poll message from the root node. This example is popular in dense networks because this approach can control network traffic effectively regardless of the network size with the cost of polling overheads.

For the UDP push example, end nodes send UDP data whenever they have data to send. Compared to the UDP poll-based approach, this technique does not require the polling overhead while it increases collision probability among transmissions of the end nodes in dense networks.

The debug_poll_demo configuration is an end-node configuration for Internet of Things (IoT) demo with TIDA-010032 data concentrator reference design.

The debug_push_leaf configuration (debug_push_leaf) was designed to provide a low power operation mode for battery-operated devices. To achieve low power consumption, this mode disables routing capability, turns RX off when idle and runs on top of the TI 15.4-Stack sleep mode. An advantage is to extend end-nodes' coverage with the built-in mesh network while keeping power consumption low. A use case will be battery-powered flow meters or in-home display connected to an electricity meter acting as a router node.

表 2 summarizes end-node configurations supported by this design.

表 2. End-Node Configuration

FEATURE	INTERMEDIATE	LEAF
Router Capability	Yes	No
RX ON when Idle	Yes	No
TI 15.4-Stack	FH non-sleep mode	FH sleep mode
Build Configuration	<ul style="list-style-type: none"> • debug_poll • debug_push • debug_poll_demo 	<ul style="list-style-type: none"> • debug_push_leaf

表 3 shows the summary of software example with CCS build configuration options. The [TIDA-01547](#), [TIDA-010024](#) and [TIDA-010032](#) reference designs are companion TI designs to run as a root node working with the end-node software examples.

表 3. CCS Build Configurations

BUILD CONFIGURATION	EXAMPLE	NOTE
debug_poll	UDP Poll	
debug_push	UDP Push	
debug_poll_demo	UDP Poll	To demo with TIDA-010032 data concentrator
debug_push_leaf	UDP Push	Low-power mode operation

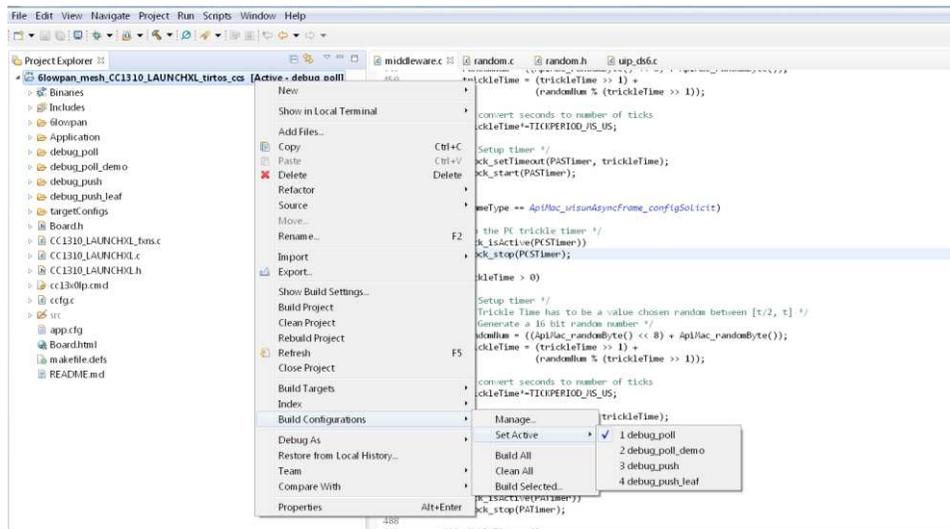


图 7. Code Composer Studio™ Build Configuration

图 8 和 图 9 显示 Code Composer Studio 屏幕截图，以显示 ARM 编译器 (TI v18.1.5.LTS)，XDC 工具 (3.51.1.18_core)，和 CC13x0 SDK 版本 (v2.40.0.20) 用于软件示例。

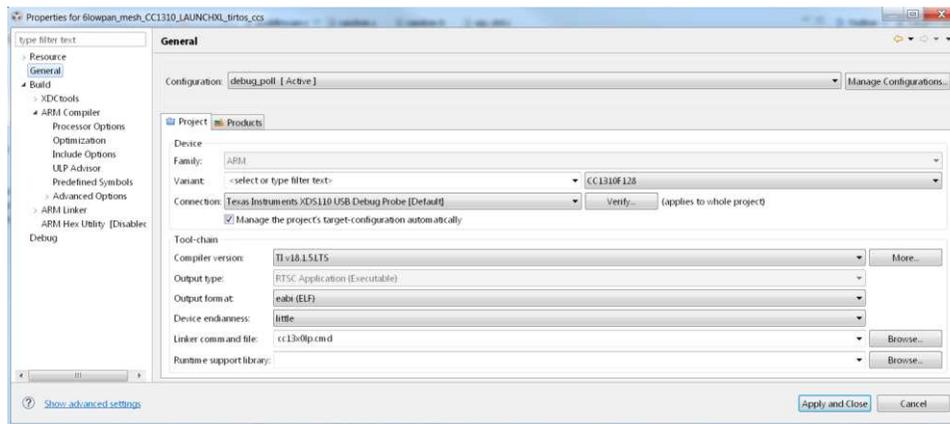


图 8. Code Composer Studio™ Compiler Version

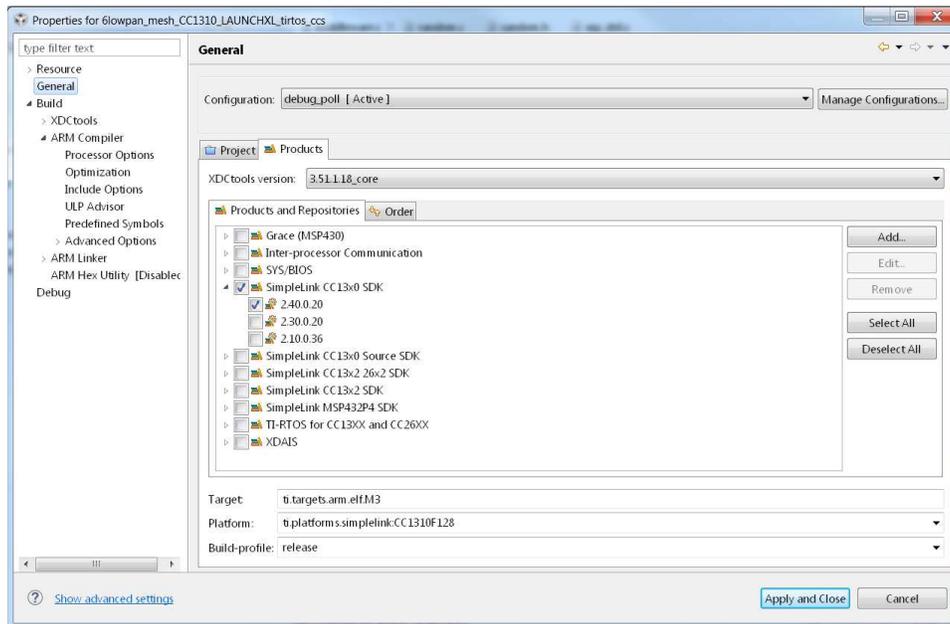


图 9. Code Composer Studio™ RTSC Screen Capture

3.1.2.2 6LoWPAN_TI_15_4_Example

This section starts with software overview, followed by details of the software architecture, and useful tips for debugging and optimizing the software.

注: This reference design provides an open-source based working example that can be a baseline software to develop end-products. The software example is not optimized in RAM or Flash usage and does not guarantee product-level quality.

3.1.2.2.1 Example Overview

This reference design implements a 6LoWPAN mesh network system working with the FH MAC over sub-1 GHz RF. The 6LoWPAN mesh network stacks run on [TI-RTOS](#), which are implemented based on CONTIKI open source.

图 10 shows the end-node example software state machine. After power on, the end node starts with *Jdllic_deviceStates_init* state and, after completing the initialization, it goes to the *Jdllic_deviceStates_fh_sync* state where the node starts FH synchronization, which consists of two steps, discussed in 节 3.1.2.2.1.1. Once the node completes the first step of the FH synchronization, the state moves to *Jdllic_deviceStates_PA_done* state. After the end-node completes the FH synchronization, it moves to *Jdllic_deviceStates_joined* state that is a ready state to start RPL route discovery. The green LED (DIO7) on the EVM indicates that the node completes the RPL-level joining process. The state machine transition has been implemented in `/Application/middleware.c`.

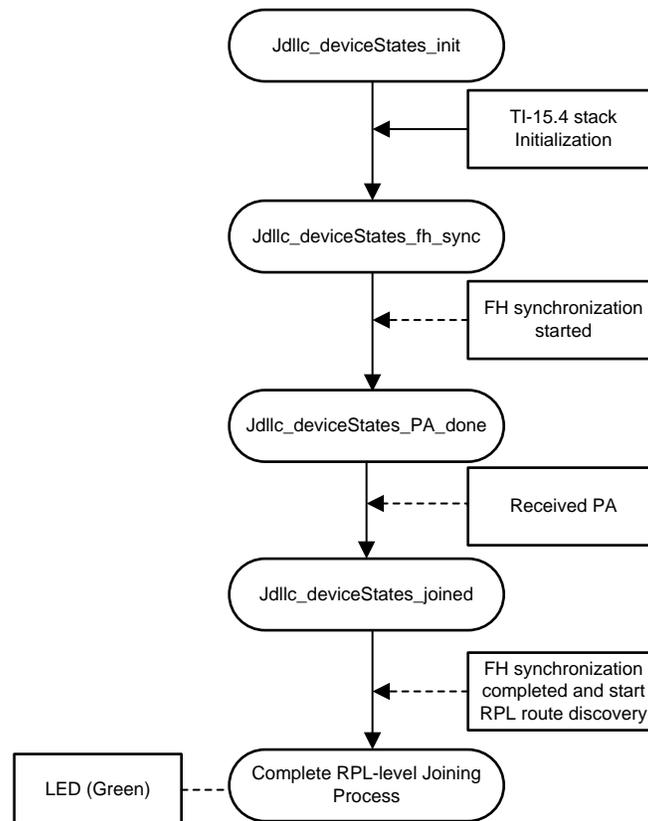


图 10. Software State Machine

3.1.2.2.1.1 FH Synchronization

FH synchronization is required to discover the FH network and to synchronize the FH timing and schedule. The underlying TI 15.4-Stack adopts WI-SUN FAN v1.0-based mechanism using four command frames:

- PAN advertisement (PA)
- PAN advertisement solicit (PAS)
- PAN configuration (PC)
- PAN configuration solicit (PCS)

FH synchronization starts with discovering neighbors, candidates of tracking parents, by performing active scan. The end nodes start with sending PAS commands at the time chosen by the trickle algorithm (RFC 6206). The PAS is sent over all the FH channels from the lowest channel number to the highest in sequence as the nodes do not know the FH timing and schedule at this time. As a response to the PAS, the PA is sent by the nodes that has already joined to the FH network. When the end nodes receive multiple PAs during the scan period (SCAN_TIMEOUT_VALUE), they will choose one of them based on the link-level metric, and then update unicast FH timing, schedule, and the PAN information with the tracking FH parent.

The next step is to send the PCS in the same way as the PAS. Once the nodes receive the PC as a response, they update the broadcast FH timing and schedule and the Group Transient Key (GTK) hash information. Receiving PC as a response of the PCS completes the FH synchronization process, which is ready to receive and send data at the network layers.

The FH timing and schedule correction is done with the received data packets that contain FH unicast and broadcast timing and schedule information elements (IEs).

注: The FH synchronization mechanism implemented in this reference design is TI proprietary and is not WI-SUN FAN standard compliant.

表 4 summarizes the trickle algorithm parameters used for FH synchronization. Depending on the network size, the parameters may need to be adjusted. These parameters are defined in /Application/middleware.h.

表 4. Trickle Algorithm Parameters

PARAMETER	VALUE	DESCRIPTION
TRICKLE_TIMEOUT_VALUE	6 seconds	Discovery trickle timer minimum timeout for PAS and PCS
TRICKLE_MAX_BACKOFF	3	Discovery trickle timer backoff exponent for PAS and PCS
SCAN_TIMEOUT_VALUE	20 seconds	Discovery trickle timer timeout for PA and PC

3.1.2.2.1.2 Keep-Alive Mechanism

The goal of the keep-alive mechanism is to detect FH sync loss throughout monitoring sync error conditions such as data transmission and reception failures. The keep-alive mechanism broadcasts keep-alive frames (a 10B TI proprietary message defined in the example) to the link-level neighbors periodically (KEEP_ALIVE_TX_TIMEOUT_VALUE) once end nodes join to the FH network. The keep-alive TX timer is reset when broadcast frames other than keep-alive frames are sent, which reduces the keep-alive traffic overheads to the network.

If end nodes do not receive broadcast frames from their tracking parents in series (KEEP_ALIVE_MAX_ATTEMPTS), FH sync loss occurs and the end nodes move immediately to the Jdllc_deviceStates_fh_sync state to restart the FH synchronization process.

In addition to the keep-alive transmissions, each node traces unicast transmission failures to the target parents. If TX attempts fail in series (MAXIMUM_NUM_DATA_FAILURE), this indicates the FH sync loss, which results in moving to the Jdllc_deviceStates_fh_sync state to re-start the FH synchronization process. 表 5 summarizes the keep-alive parameters. The keep-alive parameters are defined in /Application/middleware.h.

表 5. Keep-Alive Parameters

PARAMETER	VALUE	DESCRIPTION	COMMENTS
KEEP_ALIVE_TX_TIMEOUT_VALUE	60 seconds	Keep-alive TX interval for parents	UDP polling example only
KEEP_ALIVE_MAX_ATTEMPTS	5 times	The maximum number of failed keep-alive RX in series to indicate FH sync loss	UDP polling example only
MAXIMUM_NUM_DATA_FAILURE	5 packets	The maximum number of TX failure to indicate FH sync loss	UDP polling and UDP push examples

3.1.2.2.1.3 MAC Data Encryption

The MAC data encryption follows IEEE 802.15.4 standard with the security level options of MIC-32, MIC-64, MIC-128, ENC, ENCMIC-32, ENC-MIC-64, and ENC-MIC-128.

In the software example, the default mode is set to the security level of `ApiMac_secLevel_encMic32` and the key ID mode of `ApiMac_keyIdMode_8`, defined in the `/Application/jdllc.c`. The TI-15.4 supports the pre-shared key mechanism based on IEEE 802.15.4 standard. The security key must be pre-programmed in the software, and all the nodes in the same network must share the same pre-shared key.

Because the MAC data encryption mechanism requires a node to register its neighbors that use the data encryption, it is required to have a discovery phase throughout unsecured data exchanges. The PAS and PA frames used for the FH synchronization must be sent without data encryption. In addition, the software example uses unsecured transmissions for PC, PCS, and broadcast frames to minimize a chance that some neighbors with a better route are not detected during the discovery phase.

3.1.2.2.1.4 RPL Routing

The 6LoWPAN mesh software example uses the RPL protocol for multi-hop routing. The network formation with the RPL routing is initiated by broadcasting Destination Oriented Directed Acyclic Graph (DODAG) information object (DIO) by the root node. Once child nodes receive the DIOs, they broadcast the DIOs and send back a unicast destination advertisement object (DAO) packet to the parents that provide the best route toward the root. The DIO and DAO transmission times are determined by the trickle algorithm (RFC 6206). For the RPL metric to decide the best route, the expected transmission count (ETX) is used by default. For details on RPL routing, see the RFC standard RFC 6550 or the TI training video on [Wireless Network Challenges and Solutions for a Smarter Grid IoT](#).

3.1.2.2.1.5 6LoWPAN

The goal of the 6LoWPAN protocol is to support the IP services by reducing the gap between IPv6 and lower stacks to serve IPv6 applications on the low-end devices typically restricted in processing power, memory, and energy. The primary tasks of the 6LoWPAN are fragmentation and reassembly, IPv6 and UDP header compression, stateless IPv6 address auto-configuration, and neighbor discovery optimization. For details of the 6LoWPAN protocol, see the RFC standards RFC 4944 and RFC 6282 or the TI training video on [Wireless Network Challenges and Solutions for a Smarter Grid IoT](#).

3.1.2.2.2 Software Architecture

The 6LoWPAN mesh end-node software consists of TI 15.4-Stack, middleware layer interconnecting between TI 15.4-Stack and upper layers, 6LoWPAN mesh stacks, and the application layer. The middleware layer initializes and configures the TI 15.4-Stack and processes incoming data from the TI-15.4 or mesh network stacks. The 6LoWPAN mesh stacks cover 6LoWPAN, RPL, IPv6, ICMPv6 and UDP protocols based on CONTIKI open source. The example software provides two types of UDP applications that can be configured at compile time: UDP poll and UDP push examples.

图 11 shows the overall software architecture. The software example was implemented based on the `sensor_cc1310p` example available in the SimpleLink CC13x0 SDK v2.10.00.36. The middleware layer runs the `middleware_process` that manages the FH synchronization and keep-alive mechanism, and interfaces between TI-15.4 and mesh network stacks to handle incoming data. The middleware layer follows the same software architecture via `iCall` service defined in the `sensor_cc1310p` example. The `tcpip_task` covers the mesh network layers of 6LoWPAN, uIPv6, RPL and UDP layers. The UDP poll and push examples are parts of the `app_task` as the application layer.

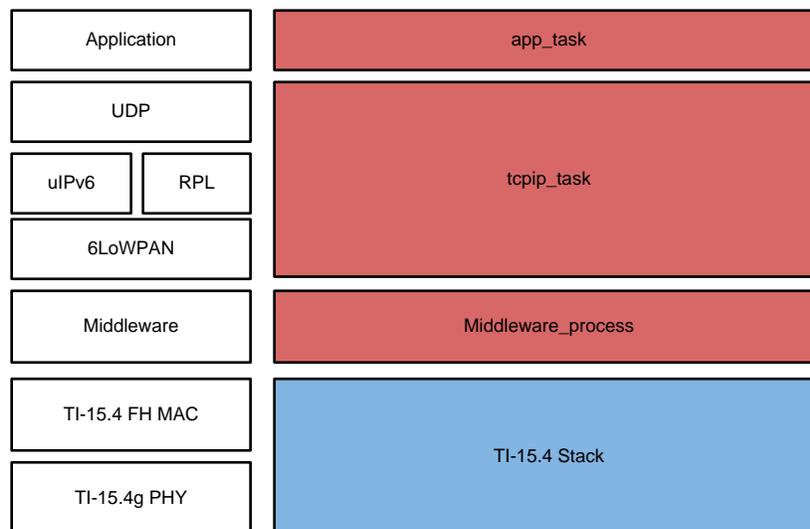


图 11. TIDA-010003 Software Architecture

3.1.2.2.2.1 TI-15.4 PHY Configuration

The TI 15.4-Stack supports multiple options for the frequency band and mode to run the FH. The software example configures the PHY mode to APIMAC_STD_US_915_PHY_1. The default configuration can be updated in the /Application/subg/config.h. The following codes list all the options for the PHY ID, which can be found in /Application/api_mac.h.

注: The TI design software example was verified with the FH operation over the PHY mode of APIMAC_STD_US_915_PHY_1.

```

    /*! PHY IDs - 915MHz US Frequency band operating mode # 1 */
    #define APIMAC_STD_US_915_PHY_1          1
    /*! 863MHz ETSI Frequency band operating mode #1 */
    #define APIMAC_STD_ETSI_863_PHY_3       3
    /*! 433MHz China Frequency band operating mode #1 */
    #define APIMAC_GENERIC_CHINA_433_PHY_128 128
    /*! PHY IDs - 915MHz LRM US Frequency band operating mode # 1 */
    #define APIMAC_GENERIC_US_LRM_915_PHY_129 129
    /*! 433MHz China LRM Frequency band operating mode #1 */
    #define APIMAC_GENERIC_CHINA_LRM_433_PHY_130 130
    /*! 863MHz ETSI LRM Frequency band operating mode #1 */
    #define APIMAC_GENERIC_ETSI_LRM_863_PHY_131 131
    /*! PHY IDs - 915MHz US Frequency band operating mode # 3 */
    #define APIMAC_GENERIC_US_915_PHY_132 132
    /*! 863MHz ETSI Frequency band operating mode #2 */
    #define APIMAC_GENERIC_ETSI_863_PHY_133 133
    
```

3.1.2.2.2.2 Middleware Layer

The Middleware layer initializes the TI 15.4-Stack, maintains MAC-level keep-alive mechanism, registers security entries to the TI 15.4-Stack, processes packets incoming from the mesh network or the TI 15.4-Stack, handles message timeout and erroneous transmissions, performs FH synchronization, and maintains the state machine which can be one of four states: *Jdllic_deviceStates_init*, *Jdllic_deviceStates_fh_sync*, *Jdllic_deviceStates_PA_done*, and *Jdllic_deviceStates_joined*.

Anend-node starts with `Jdlc_deviceStates_init` state to reset the TI 15.4-Stack, to configure initial MAC PIB and FH PIB values, and to start the PAN as end-node. The state changes to `Jdlc_deviceStates_fh_sync` state when the node initiates the FH synchronization process. As the first step of the FH synchronization, if the node receives PA as a response of the PAS command, it moves to `Jdlc_deviceStates_PA_done` state and proceeds to send the PCS command. After the FH synchronization completes by receiving PC commands, the node moves to `Jdlc_deviceStates_joined` state, which is the ready state to send data at the mesh network layers.

Similar to the `sensor_cc1310p` example available in the SimpleLink CC13x0 SDK, the communications between TI 15.4-Stack and the middleware is done via iCall messages. Conversely, the message exchanges between the mesh network stacks and the middleware layer are done via mailbox posting or direct API calls. For the details on the iCall framework, refer to `ti-15.4-stack-users-guide` in `docs/ti154stack` under the CC13x0 SimpleLink SDK installation directory.

3.1.2.2.3 Network Layers (6LoWPAN, IPv6, RPL, and UDP)

The network stacks are implemented based on the CONTIKI open source. The `tcpip_task` (in `6lowpan/uip_rpl_task.c`) processes messages incoming from the application and lower layers through mailbox. For details of the implementations, refer to the [CONTIKI open source website](#).

3.1.2.2.4 Application Layer

The example includes two types of UDP applications: UDP poll and UDP push. The `app_task` opens the UDP socket (UDP client for the end-node) with known port numbers and starts UDP data transmissions. Depending on the UDP examples, the initiator of the UDP data is different. For the UDP poll example, the root node initiates the poll message to each node to read data. For the UDP push example, each node initiates data transmissions whenever there is application data to send.

The Device Language Message Specification (DLMS) and Companion Specification for Energy Metering (COSEM) for smart metering applications use the poll-based mechanism, and the Constraint Application Protocol (CoAP) uses the mix of UDP poll (GET command) and push (periodic OBSERVE command) mechanisms. The target end-product applications can be easily integrated on top of the given UDP application.

3.1.2.2.5 LED Configuration

表 6 summarizes the LED configuration in the software example. The `board_led_type_LED1` toggles when transmission or reception events happen. The `board_led_type_LED2` turns on when end nodes join to the RPL network.

表 6. LED Configuration (CC1310)

NAME (PIN NUMBER)	EVENT	ACTION
<code>board_led_type_LED1</code> (DIO6)	Data TX/RX	TOGGLE (Red)
<code>board_led_type_LED2</code> (DIO7)	Complete RPL-level Joining	ON (Green)

3.1.2.2.3 Tips for Debugging and Optimization

This section provides useful tips with the Code Composer Studio tool to debug and optimize the software example in the end-product development phase.

3.1.2.2.3.1 Running in Debug Mode

To debug software, run the software in debug mode. The Code Composer Studio tool provides the debug mode operation. To run in debug mode with the Code Composer Studio tool:

1. Launch the target configuration for the target device (see [图 12](#)).
2. Connect the target EVM, load the program, and then run (see [图 13](#)).
3. Add global variables to debug in the *Expressions* tab (see the right top in [图 13](#)) to trace the variables in the debug mode.

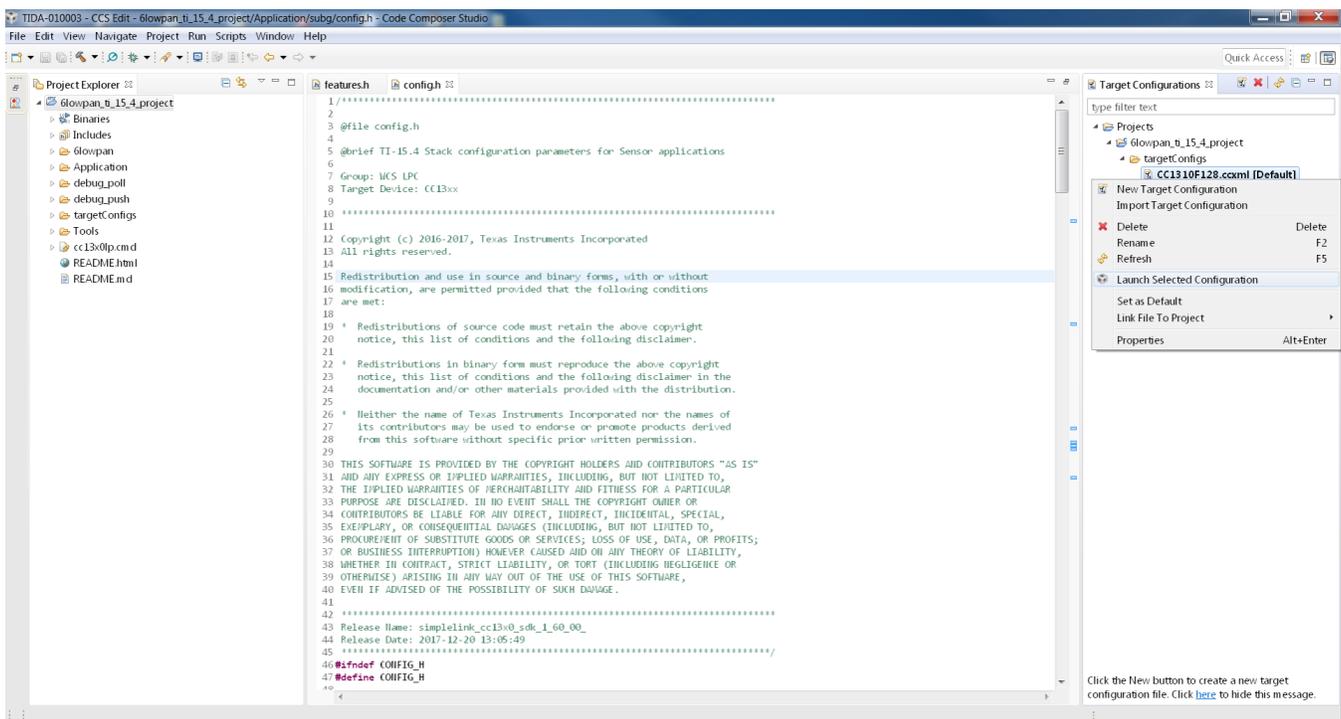


图 12. Code Composer Studio™ Debug: Launching Debug Window

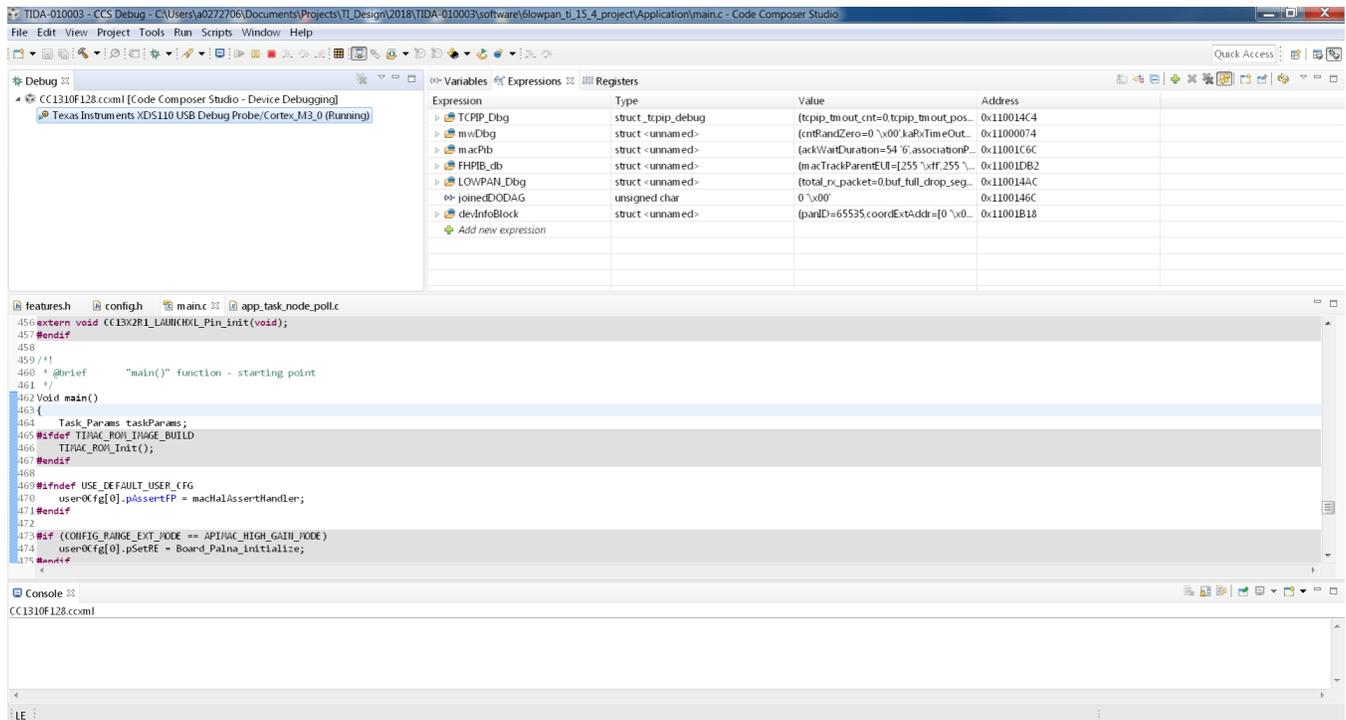


图 13. Code Composer Studio™ Debug: Running Software in Debug Mode

表 7 summarizes some global variables to debug the software example.

表 7. Global Variables for Debugging

VARIABLES	DESCRIPTION
TCPIP_Dbg	Debug counts for UDP, IPv6, ICMPv6, RPL layers
LOWPAN_Dbg	Debug counts for 6LoWPAN layer
devInfoBlock	Device state machine and PAN information
mwDbg	Debug counts for middleware layer
macPib	TI-15.4 MAC PIB
FHPIB_db	TI-15.4 FH MAC PIB
joinedDODAG	Flag to indicate RPL DODAG join state
ApiMac_extAddr	8B extended MAC address

3.1.2.2.3.2 ROV Analysis

CCS provides a useful tool to debug the software, RTOS Object View (ROV). The ROV tool helps to address various software crash issues or to optimize software examples by analyzing peak memory usage. To debug software with the ROV tool:

1. Suspend debug mode and open ROV as shown in 图 14. 图 15 shows the screen capture of ROV (on the right-bottom).
2. As an example, go to the *Detailed* and *CallStacks* taps in the *Task* menu to optimize the stack size, to trace the call stacks, or to address the stack overflow issue.

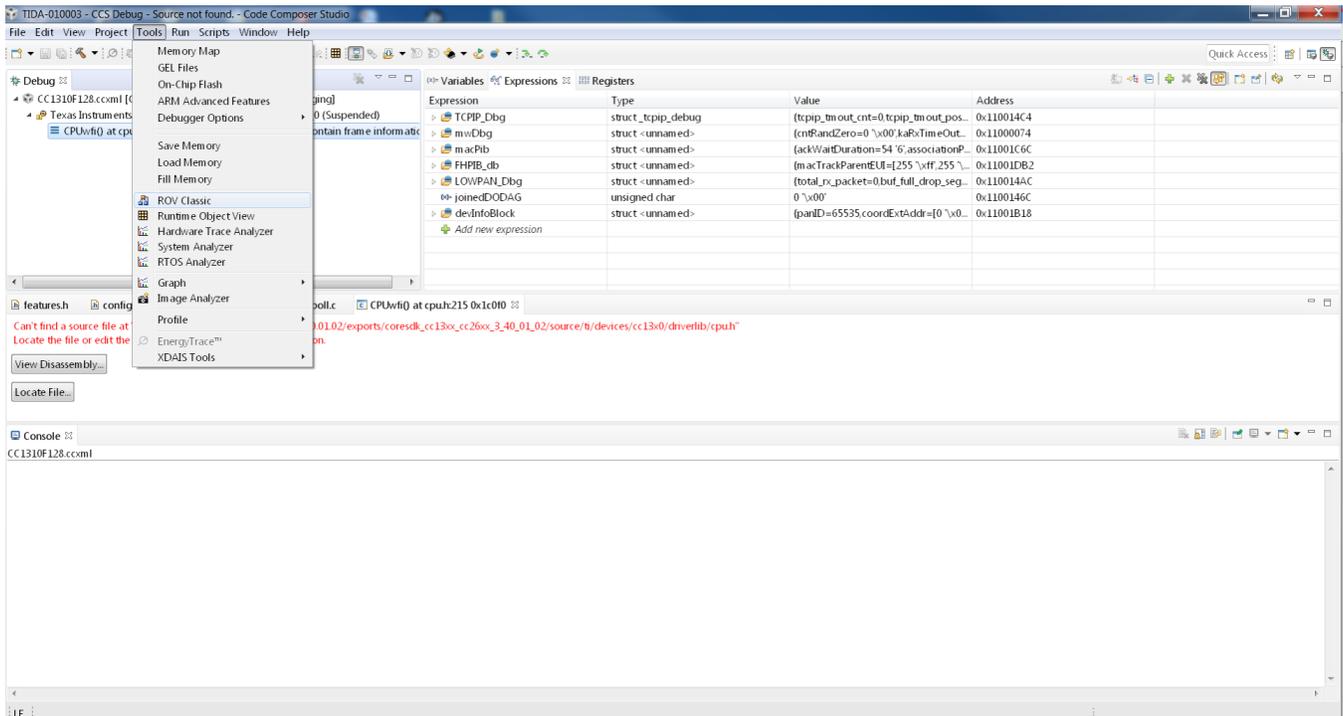


图 14. Code Composer Studio™ Debug: Launch ROV

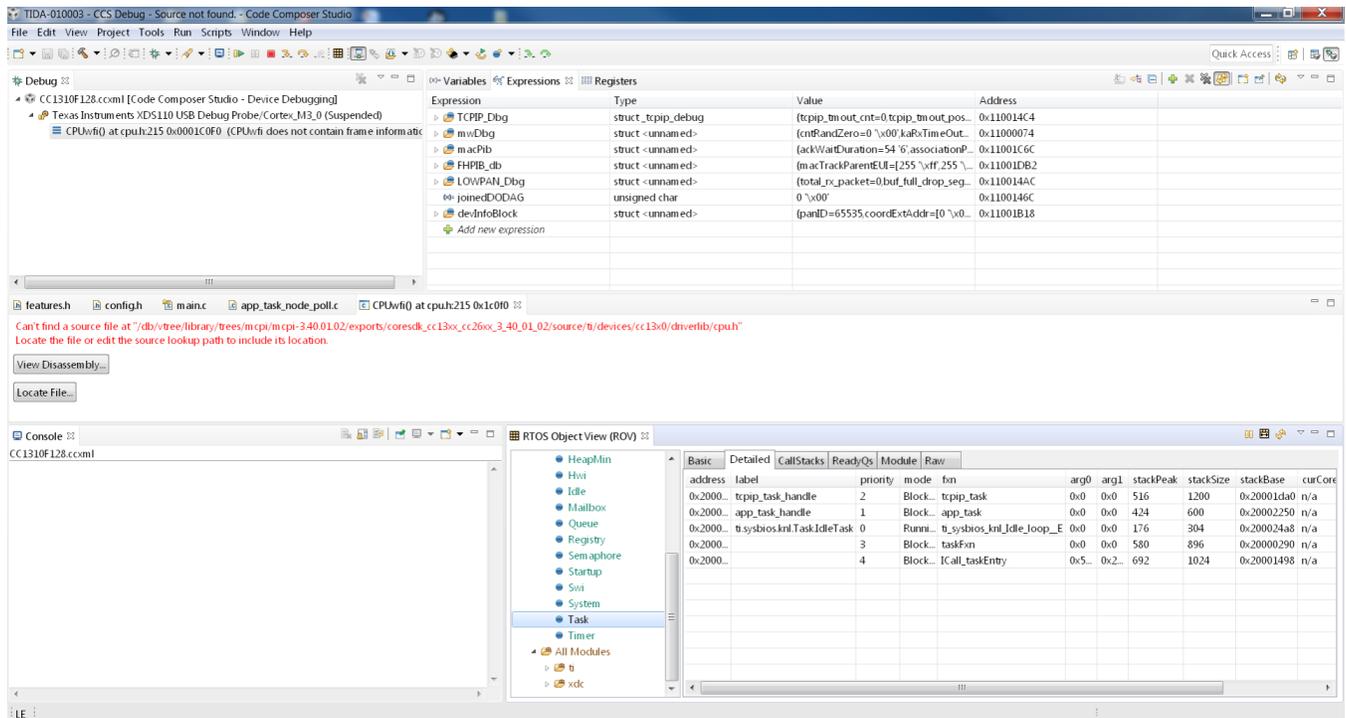


图 15. Code Composer Studio™ Debug: Debug Software with ROV Tool

3.2 Testing and Results

3.2.1 Test Setup

表 8 总结了要刷入的固件，以设置端节点和数据收集器。对于这些实验，端节点使用 CC1310 EVM 如 节 3.1.1 所示。TIDA-01547 实现了与端节点一起工作的 6LoWPAN 网状数据收集器。数据收集器软件刷入的细节在 *Flashing the Software Example* 部分中，在 [TIDA-01547 Design Guide](#) 中描述。

表 8. Firmware for Test Setup

DEVICE	CC1310	MSP432	BUILD CONFIGURATION
END NODE	6lowpan_mesh_CC1310_LAUN CHXL_tirtos_ccs.out	N/A	debug_poll
DATA COLLECTOR	coprocessor_cc13x0_lp.hex	6lowpan_mesh_example_tirtos_ccs.out	TIDA-01547 (debug_root_poll)

3.2.1.1 Creating Multi-hop Linear Topology

在小型实验室区域中创建多跳网络具有挑战性。对于实验，多跳拓扑是在软件中通过地址过滤创建的。每个节点都有一个条目列表，其中节点可以接受数据包接收。要创建线性拓扑，每个节点维护两个条目：一个是目标父节点，另一个是目标子节点。

```
// {0xC6, 0x7F, 0xA4, 0x13, 0x00, 0x4B, 0x12, 0x00} (hop-1)
uint8_t whitelist[NUM_ENTRIES][8]={
{0xF5, 0x2A, 0xF4, 0x14, 0x00, 0x4B, 0x12, 0x00},
{0xC2, 0x2F, 0xF2, 0x13, 0x00, 0x4B, 0x12, 0x00}
};
```

在软件中，此功能可以通过在 `CONFIG_MULTIHOP_TESTING` 宏中定义在 `/Application/subg/config.h` 中启用。

注： 对于测试设置，请修改在 `/Application/middleware.c` 中定义的条目列表，基于测试节点的扩展地址。对于每个节点，初始阶段设置的扩展地址存储在名为 `ApiMac_extAddr` 的全局变量中。

图 16 shows the 6-hop linear topology setup.



图 16. 6-Hop Linear Topology Setup

The multi-hop network with the address filtering technique has pros and cons in terms of network performance because a node can communicate with a dedicated set of nodes only while all the nodes can be seen with each other. A disadvantage is that a node limits the spatial reuse over multi-hop topology while an advantage is that this set-up can avoid hidden node problems.

3.2.2 Impact of Out-of-Network Interference on System Performance

The goal of these experiments is to validate the impact of external noise on the network performance. The external noise can be background channel noise and signals emitted from co-existing sub-1 GHz communication systems.

For these experiments, the UDP poll example is used over a 6-hop linear topology as shown in 图 17. The data collector sends poll messages every 5 seconds and, as a response, each node sends back the same size data packets to the data collector. To verify the impact of noise, continuous background noise is generated to the network. For details of the noise setup, see 节 3.2.2.1.

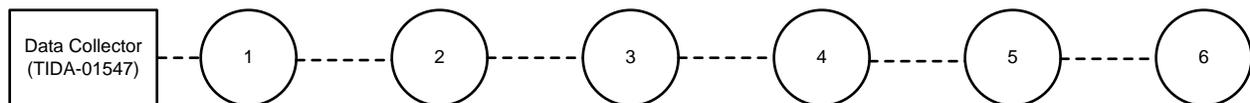


图 17. 6-hop Linear Topology

3.2.2.1 Noise Generator

The noise generator is used to verify the noise immunity feature of the FH technique in multi-hop networks. This feature is important especially when the system is running in the unlicensed bands. According to N. Baccour et al., several radio technologies such as wireless sensor networks, telemetry networks, cordless telephones, and mobile phones over the European Global System for Mobile Communications (GSM) band can cause significant in-band or out-of-band noises on the 868-MHz and 915-MHz frequency bands.

图 18 shows the SmartRF™ Studio setup to generate continuous noise with the maximum TX power of 14 dBm at a 902.2-MHz frequency [channel #0 in the FCC band (channel #0 to #128)]. Due to the maximum TX power level of the noise, the actual noise effect is not limited to the channel #0. To verify the impact of the continuous noise on the channels, 图 19 shows the noise capture in the frequency domain. As the channel space is 200 kHz, the number of channels affected by the noise generator is up to six channels (with a threshold of 0 dBm) or eight channels (with a threshold of -10 dBm) as summarized in 表 9.

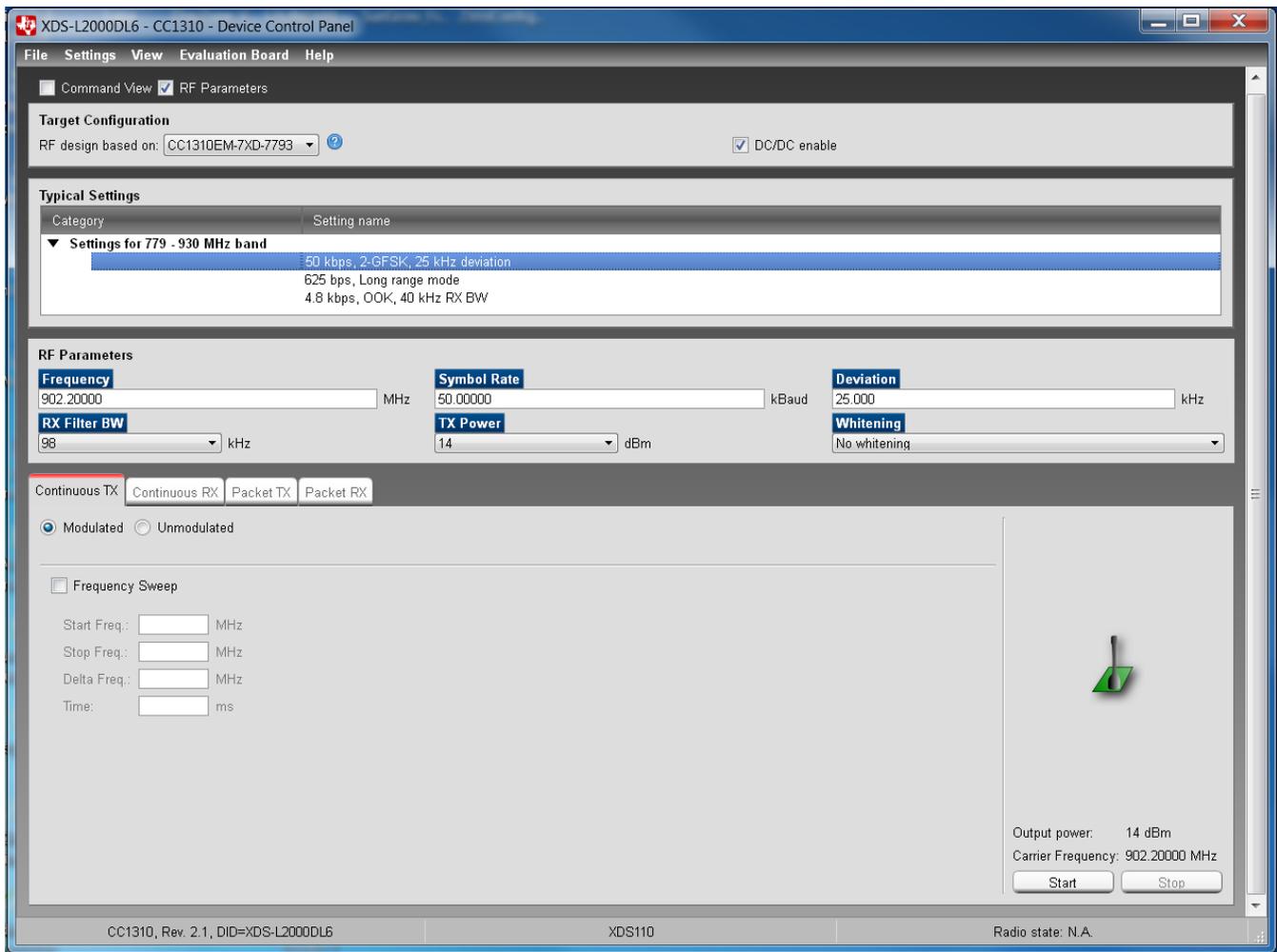


图 18. Continuous Noise at 902.2-MHz Frequency (Channel #0 in FCC Band)

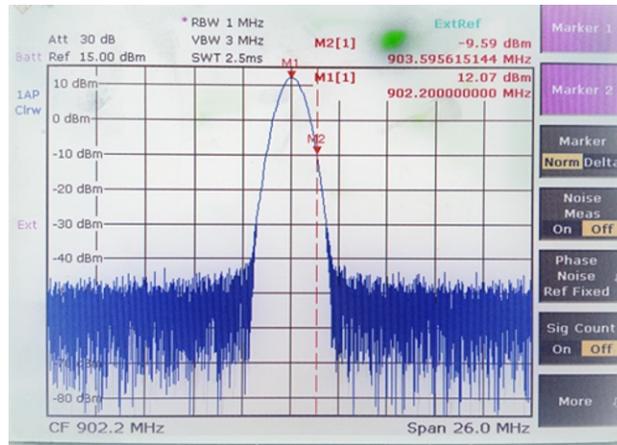


图 19. Spectrum Analyzer Capture for Continuous Noise With TX Power of 14 dBm

表 9. Number of Channels Affected by the Noise Generator

THRESHOLD	FREQUENCY	NUMBER OF CHANNELS ABOVE THE THRESHOLD
0 dBm	903.23 MHz	6
-10 dBm	903.6 MHz	8

3.2.2.2 Delivery Ratio

The delivery ratio is measured to verify the communication reliability of the FH system in noisy channels over multi-hop networks. Typically, the delivery ratio is measured in one direction based on the ratio of the number of TXs at the transmitter and the number of RXs at the receiver. In this testing setup, the delivery ratio is measured based on bidirectional communication based on the ratio of the number of TXs and the number of RXs (echo back from end nodes) at the data collector. Compared to the delivery ratio measured in one direction, the delivery ratio performance for these experiments is more conservative as the successful transmissions in both directions can increase the number of RXs.

图 20 shows the delivery ratio performance with and without continuous background noise as a function of the number of hops. The X-axis shows the number of hops and the Y-axis is delivery ratio in percentage. The result shows that the continuous background noise has no impact on the delivery ratio performance. The average delivery ratio over 6-hop networks achieves 99.7% and the worst-case performance shows 99.4% at six-hop distance.

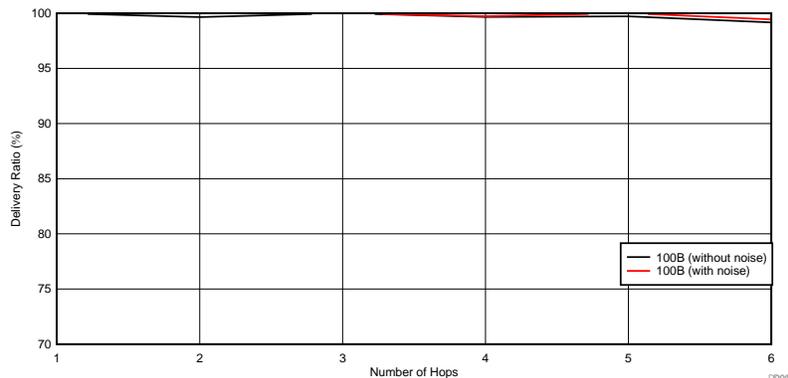


图 20. Delivery Ratio as a Function of Number of Hops

3.2.2.3 Round-Trip Time (RTT)

The RTT is measured at the application layer by calculating the time gap between the TX time of poll message sent by the root node and the RX time of data at the root node sent by an end node.

图 21 shows the RTT performance with and without the continuous background noise as a function of the number of hops. The X-axis shows the number of hops and the Y-axis shows the RTT in the unit of seconds. The experimental result shows that the RTT performance has a small gap with the continuous noise and the RTT performance gap increases as the number of hops increases. This result is explained by how the TI 15.4-Stack achieves reliable delivery ratio performance even with the continuous background noise. The TI 15.4-Stack implemented a deferred transmission mechanism to minimize the impact of noise on the delivery ratio, which increases the RTT. As the FH system uses 129 channels uniformly in the FCC band, the impact of noise on the noisy channels is negligible for the one-hop node (once every 129 transmissions). The RTT simply adds the delay for each link, and the probability to dwell on the noisy frequency channels increases as the number of hops increases. This is why the impact of noise on the RTT performance increases as the number of hops increases.

The result shows that, for one-hop distance, the average RTT performance achieves 0.19 seconds regardless of noise and, for six-hop distance, the RTT performance shows 1.17 and 1.38 seconds for the FH system without and with noise, respectively.

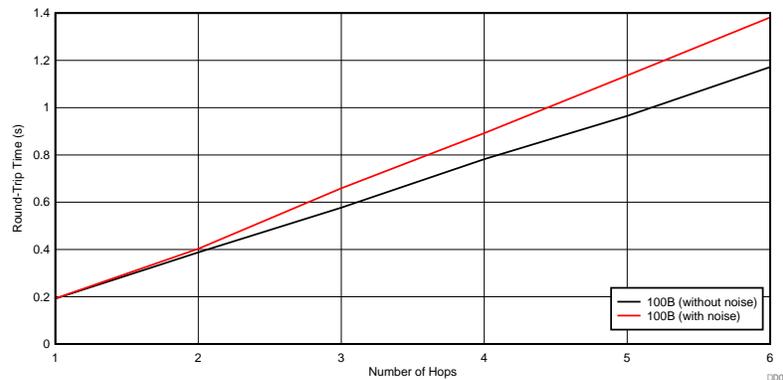


图 21. Round-Trip Time as a Function of Number of Hops

3.2.2.4 Goodput

The goodput is defined as the application-level throughput considering all the overheads in the underlying layers. The goodput performance is calculated based on the measured RTT as shown in 图 21. Because the RTT can be measured on successful transmissions only, the goodput performance does not count on the packet losses caused by channel variation or collisions. Therefore, this can be a reference on the expected goodput when data transmissions are successful.

图 22 shows the goodput performance as a function of the number of hops. The X-axis denotes the number of hops and the Y-axis shows the goodput in kbps. As expected, the goodput performance decreases as the number of hop increases due to more transmission times through multiple hops to reach at the destination. The result shows that the goodput performance varies from 8.3 kbps (100B at one-hop distance) to 1.2 kbps (100B at six-hop distance).

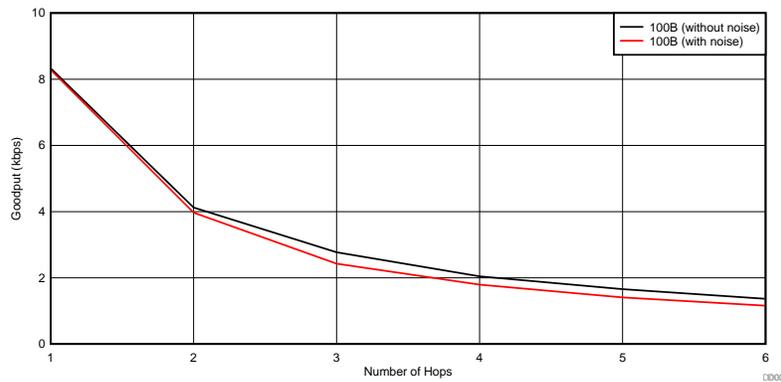


图 22. Goodput Performance as a Function of Number of Hops

3.2.3 20-Node Network Testing

The goal of this experiment is to measure delivery ratio performance as well as to verify the software reliability in a real scenario without the filtering technique to create multi-hop discussed in 节 3.2.1.1.

图 23 show the delivery ratio performance as a function of the MAC address of each node in the 20-node setup. This experiment uses the UDP polling example with 100B data, and the experiment runs for two days. The X-axis shows the MAC address of each end node and the Y-axis shows the delivery ratio in percentage. The experimental result shows that all the nodes achieve close to 100% delivery ratio performance. The average delivery ratio with 20 nodes is 99.83% and the average RTT is 0.201 second.

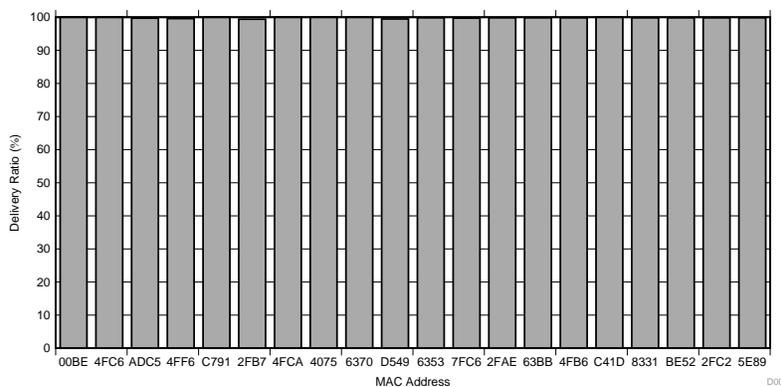


图 23. Delivery Ratio Performance With 100B UDP Polling Example

表 10 summarizes test cases to verify the software reliability and self-healing feature with the 20-node network setup. The experiments use the UDP polling example with 100B data. The test results show that the 6LoWPAN mesh software works reliability without losing connections in the 20-node setup and the keep-alive mechanism implemented in the software is functional by detecting and recovering the connection losses.

表 10. Software Reliability Test Summary

TEST CASE	DESCRIPTION	PASS or FAIL
Long-run data testing (> one week)	Run 100B UDP polling example in the 20-node setup	Pass (All 20 nodes stayed connection to the data collector)
Self-healing testing (data collector failure)	Power-cycle the data collector to see if all the 20 nodes are reconnected to the network.	Pass

表 10. Software Reliability Test Summary (continued)

TEST CASE	DESCRIPTION	PASS or FAIL
Self-healing testing (end-node failure)	Power-cycle some end-nodes to see if all the 20 nodes are reconnected to the network.	Pass

4 Design Files

4.1 Schematics

To download the schematics, see the design files at [TIDA-010003](#).

4.2 Bill of Materials

To download the bill of materials (BOM), see the design files at [TIDA-010003](#).

4.3 PCB Layout Recommendations

4.3.1 Layout Prints

To download the layer plots, see the design files at [TIDA-010003](#).

4.4 Altium Project

To download the Altium Designer® project files, see the design files at [TIDA-010003](#).

4.5 Gerber Files

To download the Gerber files, see the design files at [TIDA-010003](#).

4.6 Assembly Drawings

To download the assembly drawings, see the design files at [TIDA-010003](#).

5 Software Files

To download the software files, see the design files at [TIDA-010003](#).

6 About the Author

WONSOO KIM is a system engineer at Texas Instruments, where he is responsible for driving grid communication system solutions, defining future requirements in TI product roadmap, and providing system-level support and training focusing on communication systems for Smart Grid customers. He received a Ph.D. degree in electrical and computer engineering from the University of Texas, Austin, Texas.

修订历史记录

注：之前版本的页码可能与当前版本有所不同。

Changes from Original (June 2018) to A Revision	Page
• 已添加 向 特性 中添加了 TIDA-010024 和 TIDA-010032 作为参考设计，该设计与 TIDA-01547 完全兼容	1
• 已添加 添加了支持适用于电池供电型终端节点的低功耗模式配置的相关信息	1
• 已添加 200-kbps frequency-shift keying as a mode that is supported by TI SimpleLinkPHY for IEEE 802.15.4g	2
• 已更改 TIDA-010003 System Architecture image.....	2
• 已更改 information in Key System Specifications table	3
• 已更改 information throughout Software section	8
• 已更改 Code Composer Studio™ Build Configuration image	9
• 已更改 Code Composer Studio™ Compiler Version image	9
• 已更改 Code Composer Studio™ RTSC Screen Capture image	10
• 已更改 information in Firmware for Test Setup table.....	20
• 已更改 code in Creating Multi-hop Linear Topology section	20

重要声明和免责声明

TI“按原样”提供技术和可靠性数据（包括数据表）、设计资源（包括参考设计）、应用或其他设计建议、网络工具、安全信息和其他资源，不保证没有瑕疵且不做任何明示或暗示的担保，包括但不限于对适销性、某特定用途方面的适用性或不侵犯任何第三方知识产权的暗示担保。

这些资源可供使用 TI 产品进行设计的熟练开发人员使用。您将自行承担以下全部责任：(1) 针对您的应用选择合适的 TI 产品，(2) 设计、验证并测试您的应用，(3) 确保您的应用满足相应标准以及任何其他功能安全、信息安全、监管或其他要求。

这些资源如有变更，恕不另行通知。TI 授权您仅可将这些资源用于研发本资源所述的 TI 产品的应用。严禁对这些资源进行其他复制或展示。您无权使用任何其他 TI 知识产权或任何第三方知识产权。您应全额赔偿因在这些资源的使用中对 TI 及其代表造成的任何索赔、损害、成本、损失和债务，TI 对此概不负责。

TI 提供的产品受 [TI 的销售条款](#) 或 [ti.com](#) 上其他适用条款/TI 产品随附的其他适用条款的约束。TI 提供这些资源并不会扩展或以其他方式更改 TI 针对 TI 产品发布的适用的担保或担保免责声明。

TI 反对并拒绝您可能提出的任何其他或不同的条款。

邮寄地址：Texas Instruments, Post Office Box 655303, Dallas, Texas 75265

Copyright © 2022，德州仪器 (TI) 公司