



AN1350: Single-Band Proprietary Sub-GHz Support with OpenThread

This document describes how to configure OpenThread applications to operate on a proprietary sub-GHz band using the Silicon Labs OpenThread software development kit (SDK) and Simplicity Studio® 5 with a compatible wireless starter kit (WSTK). It also provides details on the proprietary Radio PHY supported with this feature.

KEY POINTS

- Configure OpenThread applications for proprietary sub-GHz support
- Proprietary radio PHY specifications
- Working with proprietary sub-GHz network

1 Introduction

Sub-GHz radios, like 2.4 GHz ones, can offer relatively simple wireless solutions. When compared to 2.4 GHz, sub-GHz offers several advantages, depending on the target application.

Some notable advantages of using sub-GHz radios are:

- Longer range
- Less signal fading in congested environments
- Low interference
- Low power

This application note provides a step-by-step guide to creating, building, and running an OpenThread application on a sub-GHz band.

As Thread is a 2.4GHz protocol and the specification currently does not support a sub-GHz feature, sub-GHz support has been added using:

- Proprietary radio PHY made available with the SDK and,
- Proprietary radio configurations supported by the OpenThread stack.

Note: This feature currently supports single-band operation only and so requires all the nodes in a mesh to be operating on the same sub-GHz band.

Our newest embedded software platform, the Simplicity Software Development Kit (SDK), is designed specifically for our Series 2 and Series 3 devices. The Simplicity SDK is a significant advancement in IoT development, providing our customers with a cohesive development environment for wireless innovation. It enables wireless developers to utilize advanced features and capabilities with the most recent Silicon Labs IoT devices.

Meanwhile, our Gecko Software Development Kit (GSDK) will continue to be available for users of our Series 0 and Series 1 devices. For additional information on Series 0 and Series 1 devices please reference: Series 0 and series 1 EFM32/EZR32/EFR32 device (silabs.com).

1.1 Proprietary Sub-GHz Radio PHY

For wireless applications to operate on a particular frequency band, the applications typically need the radio implementation to provide the necessary PHY support. For OpenThread applications supported on Silicon Labs platforms, the platform abstraction layer supports the 2.4GHz band by default. For the sub-GHz feature, proprietary radio PHY support has been added.

This section describes the proprietary radio PHY specifications that have been used to support this feature. The PHY specifications are compliant with the NA FCC Part 15.247 regulations.

1.1.1 Modulation Details

The proprietary radio PHY currently supported with the OpenThread SDK uses the following modulation specifications.

Table 1.1. Proprietary Sub-GHz Modulation Specification

Parameter	Configuration
Modulation	2-Level GFSK
Data Rate	500 kbps
Tx Filter BT	0.5 (Gaussian)
Modulation Index	0.76

1.1.2 Channel and Frequency Specifications

Similarly, the channel and frequency specifications for the proprietary radio PHY, and the center frequency for the supported channels have been captured in the following tables.

Table 1.2. Proprietary Sub-GHz PHY Band Parameters

Band Parameters	Value
Channel Page	23
Frequency Band (MHz)	902- 928
Channel Spacing (MHz)	1.0
Total Channels	25
Channel Numbers	0 – 24
1st Channel Center Freq. (MHz)	903.0

Table 1.3. Channels and Center Frequencies for 902-928 MHz Band

Chan. #	Fc (MHz)
0	903.0
1	904.0
...	...
23	926.0
24	927.0

1.1.3 PHR Length

The proprietary radio PHY supports 2-byte PHR. With a supported PSDU of 127 bytes, the last 7 bits of the 2nd byte represent the Frame Length.

1.1.4 Hardware Limitations

The proprietary sub-GHz feature is currently supported on radio boards supporting the 915 MHz band and using EFR32MG12 or EFR32MG13 parts only. Radio boards that could be used to test this feature are BRD4164a, BRD4170a and BRD4158a.

2 Building an OpenThread Sample App for Proprietary Sub-GHz

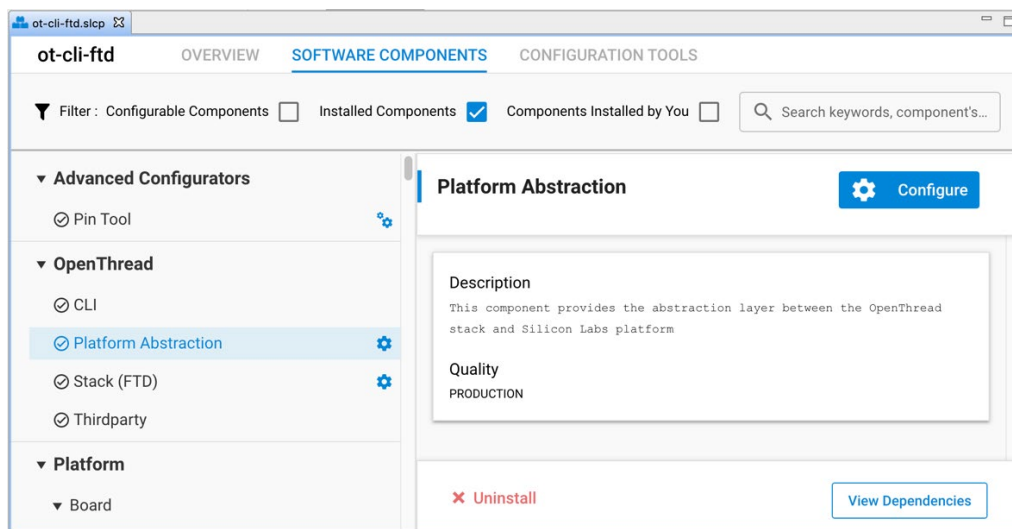
To build an OpenThread sample application for sub-GHz, you will need Simplicity Studio 5 (SSv5) and the Gecko SDK 3.2 (or higher) development environment with the OpenThread SDK package installed.

This document assumes that you have installed SSv5 and the OpenThread SDK, and that you are familiar with SSv5 and configuring, building, and flashing applications. If not, see *QSG170: Silicon Labs OpenThread Quick-Start Guide*.

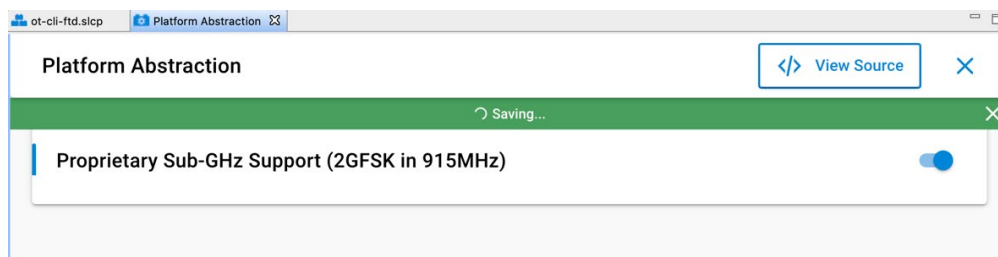
1. Connect your target development hardware (supporting proprietary sub-GHz as discussed in section [1.1.4 Hardware Limitations](#)) connected, open SSv5's File menu and select New > Silicon Labs Project Wizard. The Target, SDK, and Toolchain Selection dialog opens. Your target hardware should be populated. Click **NEXT**.
2. The Example Project Selection dialog opens. Use the Technology Type and Keyword filters to search for a specific example, for example **ot-cli-ftd**. Select it and click **NEXT**.

Note that, if you do not see the application, your connected hardware may not be compatible. To verify, in the Launcher Perspectives My Products view enter EFR32MGxx and select one of the boards. Go to the Examples tab, filter by Thread technology and verify you can see the app.

3. The Project Configuration dialog opens. Here you can rename your project, change the default project file location, and determine if you will link to or copy project files. Note that if you change any linked resource, it is changed for any other project that references it. Unless you know you want to modify SDK resources, use the default selection. Click **FINISH**. The Simplicity IDE opens with the **ot-cli-ftd** project open in the Project Configurator.
4. To configure proprietary sub-GHz support, on the SOFTWARE COMPONENTS tab, select Installed Components and select the **Platform Abstraction** component under OpenThread. Note that you can also search for a component by name in the search field.



5. Click **Configure** and enable the **Proprietary Sub-GHz Support** configuration option.



Note: If you do not see the **Configure** control or the 'Proprietary Sub-GHz Support' configuration option associated with the Platform Abstraction component, your hardware may not be compatible with the proprietary radio specifications discussed above.

6. Build the project. The generated ot-cli-ftd.s37 image may now be uploaded to your board using an SSv5 tool such as the flash programmer or Simplicity Commander.

3 Working with a Proprietary Sub-GHz Network

Silicon Labs radio boards supporting the sub-GHz ISM band are designed to operate in the US FCC 902-928 MHz band with an external whip antenna. Accordingly, when working with this feature, connect the external whip antenna using the SMA antenna connector on your radio board. For more information about this requirement, please refer to your radio board's reference manual.

3.1 Creating a Proprietary Sub-GHz Network

As mentioned in section 1 Introduction, the proprietary sub-GHz feature currently supports single band use only and so requires every node in the mesh to be running an application with the sub-GHz feature enabled. Accordingly, to create a sub-GHz network:

1. Build the **ot-cli-ftd** example with the proprietary sub-GHz feature enabled as discussed in section 2 [Building an OpenThread Sample App for Proprietary Sub-GHz](#) and flash the same application on all your nodes.
2. Use the standard OpenThread CLI commands to form and attach to a network. An example of this step is provided in Section 3.5 of *QSG170: Silicon Labs OpenThread Quick Start Guide*.
3. The resulting network formed has nodes operating on the sub-GHz band (with channels supported between 0 – 24, covering 902 – 928 MHz)

3.2 Enabling Proprietary Sub-GHz Support on an OpenThread Border Router

This section assumes that you are familiar with the basic build and install instructions for an OpenThread Border Router. If not, refer to *AN1256: Using the Silicon Labs RCP with the OpenThread Border Router*.

To enable proprietary sub-GHz support on an OpenThread Border Router:

1. Build an RCP image using Simplicity Studio 5 with the sub-GHz feature enabled. Start with the **ot-rcp** example and follow the steps described in Section 2 [Building an OpenThread Sample App for Proprietary Sub-GHz](#).
2. For the Border Router Host you can either:
 1. Use a pre-built docker image (Recommended)
<https://hub.docker.com/r/siliconlabsinc/openthread-border-router-proprietary-na-915/tags>
 2. Or, manually build the border router image for your host with the following OpenThread proprietary radio configurations set. This option requires you to modify the OT BR build scripts (details of which are beyond the scope of this document)

Configuration	Value
OPENTHREAD_CONFIG_PLATFORM_RADIO_PROPRIETARY_SUPPORT	1
OPENTHREAD_CONFIG_RADIO_2P4GHZ_OQPSK_SUPPORT	0
OPENTHREAD_CONFIG_RADIO_915MHZ_OQPSK_SUPPORT	0
OPENTHREAD_CONFIG_PLATFORM_RADIO_PROPRIETARY_CHANNEL_PAGE	23
OPENTHREAD_CONFIG_PLATFORM_RADIO_PROPRIETARY_CHANNEL_MIN	0
OPENTHREAD_CONFIG_PLATFORM_RADIO_PROPRIETARY_CHANNEL_MAX	24
OPENTHREAD_CONFIG_PLATFORM_RADIO_PROPRIETARY_CHANNEL_MASK	0x1fffffff
OPENTHREAD_CONFIG_DEFAULT_CHANNEL	0

3.3 Verifying Sub-GHz Operation

To verify if your application has been configured correctly to operate on the sub-GHz band,

1. Execute the following CLI command on your node, to retrieve the supported channel mask:

```
> channel supported
0x1fffffff
Done
```

For proprietary sub-GHz applications, the result of this command is `0x1fffffff`, indicating channels 0-24 supported for this configuration. For 2.4 GHz applications, the output returned is `0x7fff800`, indicating channels 11-26 supported for that band.

- Alternately, for a node running the sub-GHz application and that is part of an OpenThread network, you can also verify the radio information using Silicon Labs Network Analyzer.

The screenshot displays the Silicon Labs Network Analyzer interface. On the left, a network diagram shows a node with MAC address 440151928 and ID 9000. Below it, a table of transactions and a list of events are visible. The event list shows several 'MLE Advertise' events at times 277, 282, 288, and 300. The right pane shows the 'Event Detail' for the selected event at 277.554332s. The radio configuration section is expanded, showing 'Radio Config: 0x97' and 'Radio info: 0x04'. A red box highlights the 'RadioConfigId: Zigbee R23 North America 915MHz (7)'. Below this, the 'Channel Number: 4 (4)' is also highlighted. A hex dump of the captured data is shown at the bottom of the event detail pane.

Time	Dur	Summary	IPv6 Src	IPv6 Des	P#	M#	E#	Error	Warr
273...	Pac...	MLE Parent Request	32...	FFFF					
274...	Pac...	MLE Parent Request	32...	FFFF					
275...	Pac...	MLE Network Data...	32...	FFFF					
276...	Pac...	MLE Advertise	32...	FFFF					
277...	Pac...	MLE Advertise	32...	FFFF					
282...	Pac...	MLE Advertise	32...	FFFF					
288...	Pac...	MLE Advertise	32...	FFFF					
300...	Pac...	MLE Advertise	32...	FFFF					

```

Event Detail
MLE crypto: ROOT, 00 11 22 33 44 55 66 77 88 99 AA BB CC DD E
  IEEE 802.15.4 [17 bytes]
  Lowpan [3 bytes]
  Lowpan UDP [7 bytes]
  MLE Security [11 bytes]
  MLE [27 bytes]
  MLE encryption MIC [4 bytes]
  Radio Info EFR32 [7 bytes]
    Crc2: 74 61
    HW End: Tx Success (0xFD)
    RadioConfig: 0x97
      Phy Header: 2-byte phy header (1)
      RadioConfigId: Zigbee R23 North America 915MHz (7)
    Radio info: 0x04
      Antenna Select: 0x00
      Sync Word Select: 0x00
      Channel Number: 4 (4)
    Status byte: 0x02
      Error Code: Success (0)
      Protocol ID: Thread on RAIL (2)
    Info Configuration: 0x09
      TxRx Indicator: Tx (0)
      Appended info Length: 0x01
      Appended Info Version: 0x01

Hex Dump [77 bytes]
FC 18 45 41 D8 38 93 B7 FF FF F6 66 C3 3F  ..EA.8....f.?
22 F3 31 32 7F 3B 01 F0 4D 4C 4D 4C 30 26  ".12.;.MLML0&
00 15 04 00 00 00 00 00 00 00 01 C8 70 14  .....p.
47 8F 0B C3 1E 66 4D 51 CA 36 B6 32 62 80  G...fMQ.6.2b.
F8 8D F2 F6 91 D2 DE 6C 7E 4D EC 3B 8F 06  .....l-M.;..
74 61 FD 97 04 02 09  ta.....
    
```

For more details on how to capture OpenThread packets using Silicon Labs Network Analyzer, refer to section 5.4 of QSG170: *Silicon Labs OpenThread Quick Start Guide*.

Simplicity Studio

One-click access to MCU and wireless tools, documentation, software, source code libraries & more. Available for Windows, Mac and Linux!



IoT Portfolio
www.silabs.com/IoT



SW/HW
www.silabs.com/simplicity



Quality
www.silabs.com/quality



Support & Community
www.silabs.com/community

Disclaimer

Silicon Labs intends to provide customers with the latest, accurate, and in-depth documentation of all peripherals and modules available for system and software implementers using or intending to use the Silicon Labs products. Characterization data, available modules and peripherals, memory sizes and memory addresses refer to each specific device, and "Typical" parameters provided can and do vary in different applications. Application examples described herein are for illustrative purposes only. Silicon Labs reserves the right to make changes without further notice to the product information, specifications, and descriptions herein, and does not give warranties as to the accuracy or completeness of the included information. Without prior notification, Silicon Labs may update product firmware during the manufacturing process for security or reliability reasons. Such changes will not alter the specifications or the performance of the product. Silicon Labs shall have no liability for the consequences of use of the information supplied in this document. This document does not imply or expressly grant any license to design or fabricate any integrated circuits. The products are not designed or authorized to be used within any FDA Class III devices, applications for which FDA premarket approval is required or Life Support Systems without the specific written consent of Silicon Labs. A "Life Support System" is any product or system intended to support or sustain life and/or health, which, if it fails, can be reasonably expected to result in significant personal injury or death. Silicon Labs products are not designed or authorized for military applications. Silicon Labs products shall under no circumstances be used in weapons of mass destruction including (but not limited to) nuclear, biological or chemical weapons, or missiles capable of delivering such weapons. Silicon Labs disclaims all express and implied warranties and shall not be responsible or liable for any injuries or damages related to use of a Silicon Labs product in such unauthorized applications.

Note: This content may contain offensive terminology that is now obsolete. Silicon Labs is replacing these terms with inclusive language wherever possible. For more information, visit www.silabs.com/about-us/inclusive-lexicon-project

Trademark Information

Silicon Laboratories Inc.[®], Silicon Laboratories[®], Silicon Labs[®], SiLabs[®] and the Silicon Labs logo[®], Bluegiga[®], Bluegiga Logo[®], EFM[®], EFM32[®], EFR, Ember[®], Energy Micro, Energy Micro logo and combinations thereof, "the world's most energy friendly microcontrollers", Redpine Signals[®], WiSeConnect, n-Link, EZLink[®], EZRadio[®], EZRadioPRO[®], Gecko[®], Gecko OS, Gecko OS Studio, Precision32[®], Simplicity Studio[®], Telegesis, the Telegesis Logo[®], USBXpress[®], Zentri, the Zentri logo and Zentri DMS, Z-Wave[®], and others are trademarks or registered trademarks of Silicon Labs. ARM, CORTEX, Cortex-M3 and THUMB are trademarks or registered trademarks of ARM Holdings. Keil is a registered trademark of ARM Limited. Wi-Fi is a registered trademark of the Wi-Fi Alliance. All other products or brand names mentioned herein are trademarks of their respective holders.



Silicon Laboratories Inc.
400 West Cesar Chavez
Austin, TX 78701
USA

www.silabs.com