



# AN1017: Zigbee<sup>®</sup> and Thread Coexistence with Wi-Fi

---

This application note describes the impact of Wi-Fi on zigbee and Thread and methods to improve coexistence with Wi-Fi. First, design considerations to improve coexistence without direct interaction between zigbee/Thread and Wi-Fi radios are described. These techniques are applicable both to the EM35x/EM358x and Mighty Gecko family (EFR32MG). Next, Silicon Lab's Packet Traffic Arbitration (PTA) support to coordinate 2.4 GHz RF traffic for co-located zigbee/Thread and Wi-Fi radios is described. This software is available for the EFR32MG only.

Additional details about the implementation of managed coexistence is available in an expanded version of this application note, *AN1017-NDA: ZigBee and Thread Coexistence with Wi-Fi*, available under non-disclosure from Silicon Labs technical support.

## KEY FEATURES

- Wi-Fi Impact on zigbee/Thread
- Improving unmanaged coexistence
- Implementing managed coexistence

## 1 Introduction

The 2.4 GHz ISM band supports Wi-Fi (IEEE 802.11b/g/n), zigbee/Thread (IEEE 802.15.4), *Bluetooth®*, and Bluetooth low energy. The simultaneous and co-located operation of these different 2.4 GHz radio standards can degrade performance of one or more of the radios. To improve interference robustness, each of the 2.4 GHz ISM radio standards support some level of collision avoidance and/or message retry capability. At low data throughput rates, low power levels, and/or sufficient physical separation, these 2.4 GHz ISM standards can co-exist without significant performance impacts. However, recent customer trends are making coexistence more difficult:

- Increased Wi-Fi transmit power level for “extended range”
  - +30 dBm Wi-Fi Access Points are now common.
- Increased Wi-Fi throughput
  - Depending on achievable SNR, high throughput requirements for file transfers and/or video streaming may result in high Wi-Fi duty cycle within 2.4 GHz ISM band.
- Integrating Wi-Fi, zigbee, Thread, and Bluetooth low energy into the same device for gateway functionality
  - This is required by Home Automation and Security applications, and provides easier end-node commissioning using Bluetooth low energy.

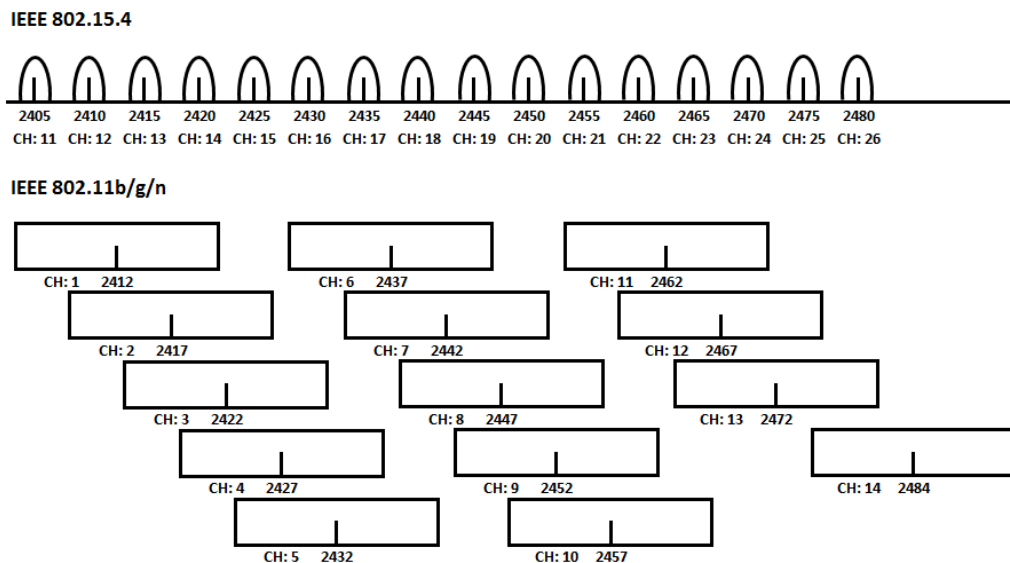
This application note describes the impact of Wi-Fi on zigbee and Thread and methods to improve coexistence with Wi-Fi on two Silicon Labs integrated circuits, the EM35x/EM358x and the Mighty Gecko family (EFR32MG).

- The section **Unmanaged Coexistence** describes design considerations to improve coexistence without direct interaction between zigbee/Thread and Wi-Fi radios.
- The section **Managed Coexistence** describes Silicon Lab's PTA (Packet Traffic Arbitration) support to coordinate 2.4 GHz RF traffic for co-located zigbee/Thread and Wi-Fi radios. PTA support is available for the EFR32MG only.

**Note:** Current Bluetooth and Bluetooth low energy solutions operate at less than +10 dBm transmit power level and implement automatic frequency hopping (AFH). Even with Bluetooth at +10 dBm, Silicon Labs' testing indicates minimal impact from co-located Bluetooth on 802.15.4 performance. 802.15.4 device join success remains at 100% and 802.15.4 message loss remains at 0%. 802.15.4 MAC retries increase slightly due to occasional co-channel and adjacent channel collisions between Bluetooth AFH and fixed 802.15.2 channel. However, the Wi-Fi coexistence discussion and solutions described in this application note can be applied equally well to Bluetooth coexistence. As such, all following solutions presented for “Wi-Fi Coexistence” can be applied to “Wi-Fi/Bluetooth Coexistence”.

## 2 Wi-Fi Impact on Zigbee/Thread

World-wide, Wi-Fi (IEEE 802.11b/g/n) supports up to 14 overlapping 20/22 MHz bandwidth channels across the 2.4 GHz ISM band with transmit power levels up to +30 dBm. Similarly, 2.4 GHz zigbee and Thread (based on IEEE 802.15.4) support 16 non-overlapping 2 MHz bandwidth channels at 5 MHz spacing with transmit powers up to +20 dBm. These Wi-Fi and zigbee/Thread channel mappings are shown in the following figure.



**Figure 1. 802.15.4 and 802.11b/g/n Channel Mapping (World-Wide)**

Actual channels available vary by country. For example, in the USA, Wi-Fi channels 1 through 11 are available and zigbee channels 11 through 26 are available, although channels 25 and 26 require reduced transmit power levels to meet FCC requirements (North America only).

To better understand the effects of Wi-Fi on zigbee/Thread, Silicon Labs' measured the impact of a 100% duty-cycled 802.11n (MCS3, 20 MHz bandwidth) blocker transmitting at various power levels while receiving an 802.15.4 message transmitted at various power levels. The results for co-channel, adjacent channel, and "far-away" channel are shown in the following three figures. All 802.11n and 802.15.4 power levels are referenced to the Silicon Labs' Wireless Gecko (EFR32MG1) RF input. The test application was developed using Silicon Labs' EmberZNet PRO (zigbee) stack with NodeTest running on the EFR32 DUT (Device Under Test) and a test script to control the DUT and RF test equipment. Since this is an 802.15.4 focused test, results are identical for Wi-Fi blocking Thread.

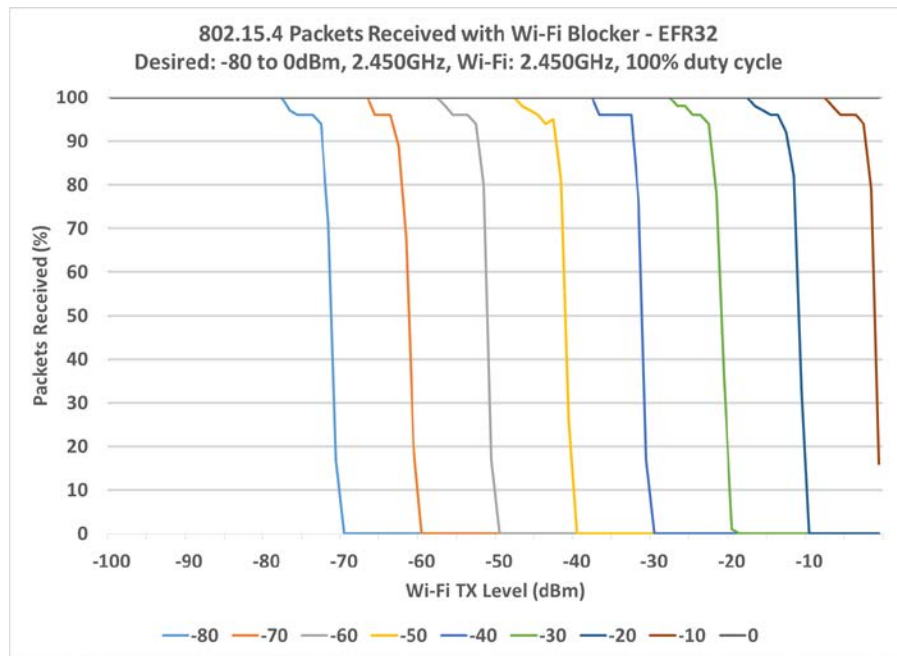


Figure 2. 100% Duty Cycled 802.11n Blocker with Desired 802.15.4 at Co-Channel

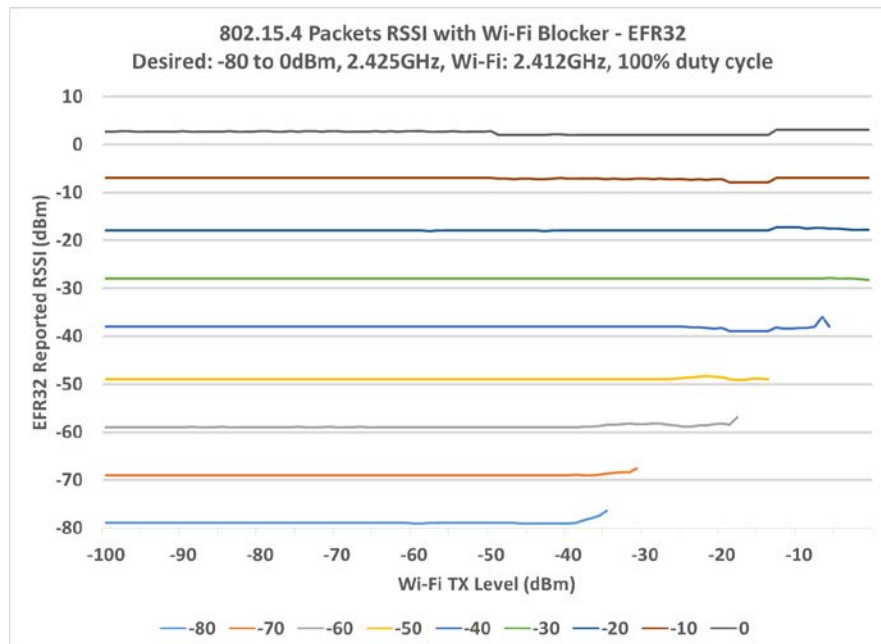
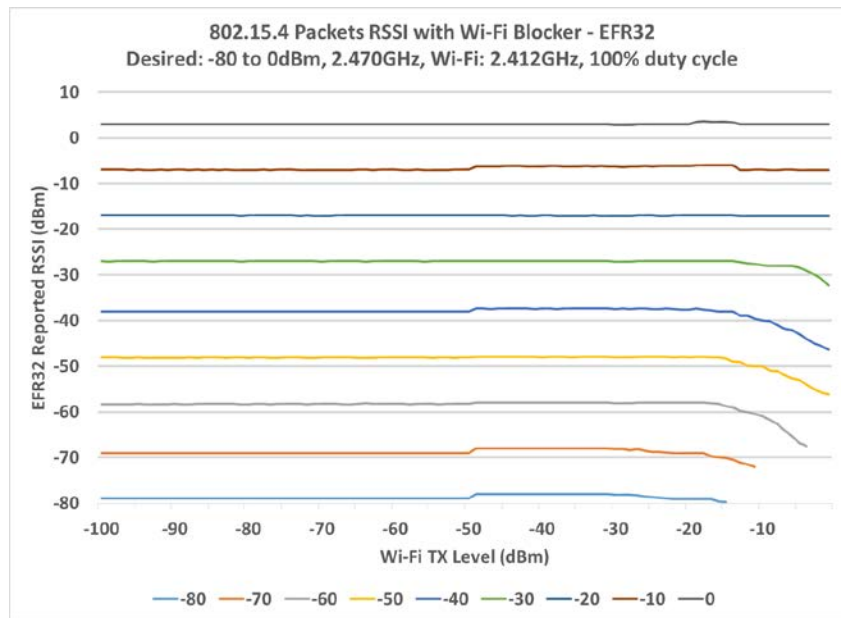


Figure 3. 100% Duty Cycled 802.11n Blocker with Desired 802.15.4 at Adjacent Channel



**Figure 4. 100% Duty Cycled 802.11n Blocker with Desired 802.15.4 at “Far-Away” Channel**

From these three figures, as well as other measurements using the EM35x/EM358x (not shown), the key observations about the impact of Wi-Fi on zigbee/Thread are:

#### Co-Channel:

- EFR32MG1 can receive an 802.15.4 signal down to 6 dB weaker than aggregate Wi-Fi transmit power (100% duty cycle).
- EM35x/EM358x with and without FEM (front-end module) can receive an 802.15.4 signal down to 6 dB weaker than aggregate Wi-Fi transmit power (100% duty cycle).
- 802.15.4 transmits can also be blocked by Wi-Fi transmit power tripping the 802.15.4 -75 dBm Clear Channel Assessment (CCA) threshold.

#### Adjacent Channel:

- EFR32MG1 can receive a -80 dBm 802.15.4 signal with -35 dBm or weaker Wi-Fi transmit power (100% duty cycle).
- EM35x/EM358x without FEM can receive a -80 dBm 802.15.4 signal with -38 dBm or weaker Wi-Fi transmit power (100% duty cycle), -43 dBm or weaker with Skyworks SE2432L FEM LNA (Low Noise Amplifier) enabled.

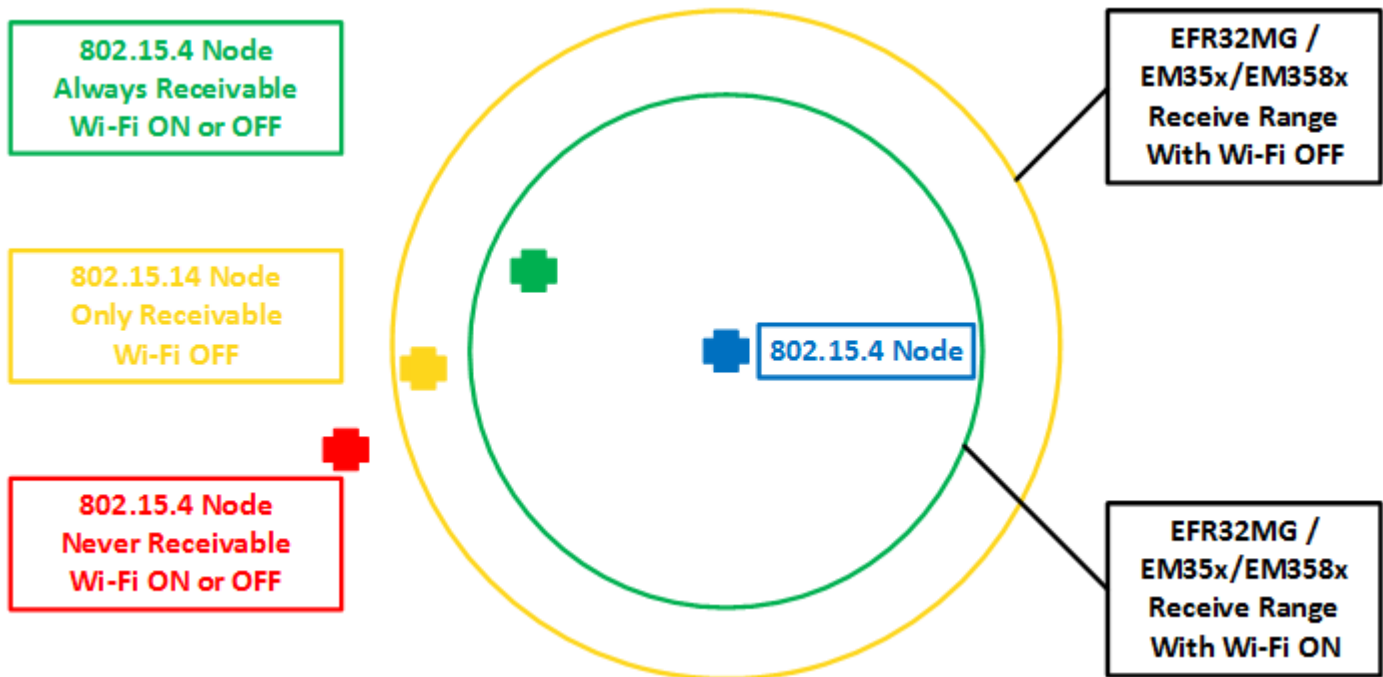
#### “Far-Away” Channel:

- EFR32MG1 can receive a -80 dBm 802.15.4 signal with -15 dBm or weaker Wi-Fi transmit power (100% duty cycle).
- EM35x/EM358x without FEM can receive a -80 dBm 802.15.4 signal with -22 dBm or weaker 100% Wi-Fi transmit power (100% duty cycle), -27 dBm or weaker with Skyworks SE2432L FEM LNA enabled.

In a real world environment, Wi-Fi is typically not 100% duty cycle and only approaches 100% duty cycle during file transfers or video stream in low Wi-Fi SNR conditions. In the above three figures, the EFR32MG1 (or EM35x/EM358x) receive sensitivity varies as the Wi-Fi blocker turns ON/OFF. The net result is the ability to see weaker signals when Wi-Fi is OFF, but not when strong Wi-Fi is ON (actively transmitting).

The following figure illustrates the receive range of a node (blue node) near a strong Wi-Fi transmitter. Relative to the blue 802.15.4 node, the area inside the green circle represents the receive range when Wi-Fi is ON. The area between the green and yellow circles represents the receive range when Wi-Fi is OFF. From this figure:

- The green node is always receivable by the blue node.
- The yellow node is only receivable by the blue node when Wi-Fi is OFF.
- The red node is never receivable by the blue node.
- The yellow and red nodes are always receivable by the green node.



**Figure 5. EFR32MG1 or EM35x/EM358x Receiver Desensitized when Wi-Fi Transmitting**

Depending on each node's type (coordinator, router, or end device) and the Wi-Fi duty cycle, the impact of strong Wi-Fi turning ON/OFF will vary.

In a zigbee network:

- Coordinator: Tasked with network creation, the control of network parameters, and basic maintenance, in addition to performing an application function, such as aggregating data or serving as a central control point or gateway.
- Router: In addition to running an application function, a Router can receive and retransmit data from other nodes.
- End Device Typically a battery-powered device (Sleepy End Device) running an application function and able to talk to a single parent node (either the Coordinator or a Router). End Devices cannot relay data from other nodes.

In a Thread network:

- Border Router: Provides Thread Network node connectivity to other devices in external networks (e.g., Internet access).
- Router: In addition to running an application function, a Router can receive and retransmit data from other nodes and provide join and security capability. When routing function is not needed by network, a Router can downgrade to a Router-Eligible End Device.
- Router-Eligible End Device: In addition to running an application function, a Router-Eligible End Device can receive and retransmit data from other nodes. When additional routers needed by network, a Router-Eligible End Device can upgrade to a Router.
- Sleepy End Device: Typically a battery-powered device running an application function and able to talk to a single parent node (either a Border Router, Router, or Router-Eligible End Device). Sleepy End Devices cannot relay data from other nodes.

Two zigbee cases are considered below, but many other cases are possible.

#### **Case 1: Zigbee Coordinator near strong Wi-Fi plus three end-nodes**

For this case, Figure 5 is composed of:

1. Coordinator: Blue node
2. End Devices: Green, Yellow, and Red nodes

In this simple network, each end device attempts to join the network formed by the coordinator. However, the red node is outside of receive range and cannot join. With Wi-Fi OFF, both the green and yellow nodes successfully join the network and have no issues sending messages to the coordinator. Regardless of Wi-Fi ON/OFF duty cycle, the green node remains successful sending messages to the coordinator.

With Wi-Fi ON/OFF at low-duty cycle, some messages from the yellow node are periodically blocked, but zigbee retry mechanisms are effective in getting the messages to the coordinator. However, with Wi-Fi ON/OFF at high-duty cycle, many messages from the yellow node are blocked and zigbee retry mechanisms may be exhausted. Even when retry mechanisms are successful, the message latency increases. If the yellow node is a battery powered sleepy end device, it must remain active longer to execute retries, reducing battery life.

**Case 2: Zigbee Coordinator near strong Wi-Fi, router within always receive range, plus two end-nodes**

For this case, Figure 5 is composed of:

1. Coordinator: Blue node
2. Router: Green node
3. End Devices: Yellow and Red nodes

In this simple network, the green router forms a route directly to the coordinator, maintained regardless of Wi-Fi ON/OFF duty cycle. With Wi-Fi OFF, the yellow node forms a route directly to the blue coordinator at a lower route cost than a route via the green router. The red node cannot be received by the coordinator and its messages are also routed through the router to the coordinator.

With Wi-Fi OFF, the green router, the yellow node, and the red node (via the green router) have no issues sending messages to the coordinator. Regardless of Wi-Fi ON/OFF duty cycle, the green router and the red node (via the green router) remain successful sending messages to the coordinator. With Wi-Fi ON/OFF at low-duty cycle, some messages from the yellow node are periodically blocked, but zigbee retry mechanisms are effective in getting the messages to the coordinator.

With Wi-Fi ON/OFF at high-duty cycle, many messages from the yellow node are blocked and zigbee retry mechanisms may be exhausted. If Wi-Fi ON/OFF stays at high-duty cycle for sufficient time, the network responds by restructuring the yellow node to route messages to the coordinator via the router. However, this route rediscover takes time and messages may be lost. As long as Wi-Fi ON/OFF remains high-duty cycle, the yellow node messages will continue to go through the router, which forwards messages to the coordinator.

However, when Wi-Fi ON/OFF returns to low-duty cycle, the network will, due to lower route cost, return to the original structure with the yellow node sending messages directly to the coordinator.

Under conditions with Wi-Fi ON/OFF switching between low and high duty cycles, the network may switch back and forth between these two route states. During these switching events, messages from the yellow end-node to the coordinator are lost.

## 3 Unmanaged Coexistence

The unmanaged coexistence recommendations that follow provide guidance on how to maximize the EFR32MG1 or EM35x/EM358x message success with strong nearby Wi-Fi.

### 3.1 Implement Frequency Separation

From the observations in the previous section, co-channel operation of 802.15.4 with 100% duty cycle Wi-Fi blocks most of the 802.15.4 messages and must be avoided. Also, EFR32MG1 tolerates a 20 dB stronger Wi-Fi signal in “far-away” channel case than in adjacent channel case. The 802.15.4 network performance is improved by maximizing the frequency separation between the Wi-Fi network and the 802.15.4 network.

If the Wi-Fi and 802.15.4 radios are implemented with a common host (MCU controlling both radios), then the host should attempt to maximize the frequency separation. For Wi-Fi networks, the access point (AP) establishes the initial channel and, in auto channel configuration, is free to move the network to another channel using the Channel Switch Announcement, introduced in 802.11h, to schedule the channel change.

For Thread networks, frequency separation implementation depends on the application layer. For zigbee networks, the coordinator establishes the initial channel. However, when implemented by the product designer, a Network Manager function, which can be on the coordinator or on a router, can solicit energy scans from the mesh network nodes and initiate a network channel change as necessary to a quieter channel.

**Note:** The Network Manager function is not a mandatory feature, but rather must be implemented using tools/functions provided by the stack.

Details on mesh network channel frequency agility can be found in:

- <http://community.silabs.com/t5/Wireless-Knowledge-Base/How-does-Frequency-Agility-work-in-the-EmberZNet-PRO-stack/ta-p/113319>
- <http://community.silabs.com/t5/Wireless-Knowledge-Base/How-can-I-test-frequency-agility/ta-p/113295>

### 3.2 Operate Wi-Fi with 20 MHz Bandwidth

Since Wi-Fi 802.11n uses OFDM sub-carriers, third-order distortion products from these sub-carriers extend one bandwidth on each side of the Wi-Fi channel. 802.11n can operate in 20 MHz or 40 MHz modes. If operated in 40 MHz mode, 40 MHz of the 80 MHz ISM band is consumed by the Wi-Fi channel. However, an additional 40 MHz on each side can be affected by third-order distortion products. These third-order products can block the 802.15.4 receiver and is the primary reason adjacent channel performance is 20 dB worse than “far-away” channel performance.

In proposing 40 MHz mode for 802.11n, the Wi-Fi standard anticipated potential issues with other 2.4 GHz ISM devices when Wi-Fi operated in 40 MHz mode. During association, any Wi-Fi station can set the **Forty MHz Intolerant** bit in the HT Capabilities Information. This bit informs the Wi-Fi access point that other 2.4 GHz ISM devices are present, forcing the entire Wi-Fi network to 20 MHz mode.

If the Wi-Fi and 802.15.4 radios are implemented with a common host, then the host should have the Wi-Fi radio set the **Forty MHz Intolerant** bit during association to force the Wi-Fi to 20 MHz mode, improving the 802.15.4 performance.

If the application requires Wi-Fi to operate in 40 MHz mode, frequency separation must be maximized by placing Wi-Fi channels and 802.15.4 channel at opposite ends of the 2.4 GHz ISM band.

From Silicon Labs’ managed coexistence testing, 802.15.4 performance with 40MHz Wi-Fi, for the same Wi-Fi RF duty cycle, is comparable to 802.15.4 performance with 20MHz Wi-Fi. While 802.15.4 performance with 100% Wi-Fi RF duty cycle is inherently impaired, 40MHz Wi-Fi, for the same target Wi-Fi data rate, has a lower RF duty cycle than 20MHz Wi-Fi, providing the 802.15.4 radio more frequent and longer time gaps for successful transmits and receives.

### 3.3 Increase Antenna Isolation

Also, from the observations in section **Wi-Fi Impact on Zigbee/Thread**, minimizing the Wi-Fi energy seen by the 802.15.4 RF input improves the 802.15.4 receive range. For example, in the “far-away” channel case with 100% Wi-Fi duty cycle, a -80 dBm 802.15.4 message can be received when the Wi-Fi energy at EFR32MG1 input is -15 dBm or less. If the Wi-Fi transmit power level is +10 dBm, 25 dB or more antenna isolation between the Wi-Fi transmitter and 802.15.4 RF input is sufficient to always receive a -80 dBm 802.15.4 signal, Wi-Fi ON or OFF.



Increased antenna isolation can be achieved by:

- Increasing the distance between antennas. In open-space, far-field, power received is proportional to  $1/R^2$ , where R is distance between antennas.
- Taking advantage of antenna directionality. A monopole antenna provides a null along the axis of the antenna, which can be directed toward the Wi-Fi antenna(s).

### 3.4 Use Zigbee/Thread Retry Mechanisms

The 802.15.4 specification requires retries at the MAC layer, which are implemented in Silicon Labs' EmberZNet PRO stack. To further improve message delivery robustness, Silicon Labs EmberZNet PRO stacks also implements NWK retries, wrapping the MAC retries. The user application can also take advantage of APS retries, wrapping the NWK retries. More information on the retry mechanisms can be found at:

- <http://www.silabs.com/support%20documents/technicaldocs/ug103.2.pdf> (Section 4.4)
- <http://community.silabs.com/t5/Wireless-Knowledge-Base/How-does-the-EmberZNet-stack-retry-work/ta-p/113287>

These retry mechanisms are effective at improving message delivery. However, under high interference conditions, message latency increases.

For Thread networks, 802.15.4 retries at the MAC layer still apply. However, other message retry mechanisms depend on the application layer.

### 3.5 Remove FEM (or operate FEM LNA in Bypass)

EFR32MG1 can deliver nearly +20 dBm transmit power and has excellent receiver sensitivity without an external FEM. However, many EM35x/EM358x applications utilize an external FEM to increase transmit power to +20 dBm for increased range (in regions where this is permitted, e.g. the Americas). The additional FEM LNA receive gain also improves sensitivity. However, this additional gain also degrades the EM35x/EM358x linearity performance in the presence of strong Wi-Fi.

As an example, a Skyworks' SE2432L FEM combined with an EM35x/EM358x provides increased transmit power and, when no blockers are present, increased receiver sensitivity. However, in the presence of strong 100% duty cycle Wi-Fi, EM57x/EM358x with SE2432L shows 5 dB degradation in receiver sensitivity when compared to EM57x/EM358x only, for both adjacent and "far-away" channel cases.

For best receive sensitivity in the presence of strong Wi-Fi blockers, either eliminate the FEM or operate the FEM LNA in bypass mode. This recommendation is a trade-off as receive sensitivity without Wi-Fi blockers is improved with FEM LNA gain enabled.

## 4 Managed Coexistence

The market trends of higher Wi-Fi transmit power, higher Wi-Fi throughput, and integration of Wi-Fi and 802.15.4 radios into same device has the following impacts:

- Advantages:
  - Host can implement frequency separation between Wi-Fi and 802.15.4.
  - Co-located Wi-Fi can force Wi-Fi network to 20 MHz bandwidth.
  - Wi-Fi and 802.15.4 radios can communicate on 2.4 GHz ISM (Industrial, Scientific, and Medical) transmits and receives.
- Disadvantages:
  - Higher Wi-Fi transmit power requires greater antenna isolation.
  - Higher Wi-Fi throughput results in higher Wi-Fi duty cycle.
  - Antenna isolation is limited by device size (only 15-20 dB isolation is not unusual).

Assuming frequency separation achieves the “far-away” channel case and Wi-Fi only uses 20 MHz bandwidth, a +30 dBm Wi-Fi transmit power level at 100% duty cycle requires 45 dB antenna isolation to receive -80 dBm 802.15.4 messages. This is generally not achievable in small devices with co-located Wi-Fi and 802.15.4.

Managed Coexistence takes advantage of communication between the co-located Wi-Fi and 802.15.4 radios to coordinate each radio's access to the 2.4 GHz ISM band for transmit and receive. For the EFR32, Silicon Labs has implemented a coordination scheme compatible with Wi-Fi devices supporting PTA (Packet Traffic Arbitration). This PTA-based coordination allows the EFR32 to signal the Wi-Fi when receiving a message or wanting to transmit a message. When the Wi-Fi device is made aware of the EFR32 requiring the 2.4 GHz ISM band, any Wi-Fi transmit can be delayed, improving zigbee/Thread message reliability.

The first section discusses PTA support hardware options, and the second discusses PTA support software setup.

### Notes:

1. PTA support software is only available for EFR32MG1 under EmberZNet PRO 5.8.0 or later (zigbee) and in Silicon Labs Thread 2.1.0 or later (Thread).
2. While PTA support software is not available for EM3x, EM3x does support the RHO (Radio Hold OFF) and TXA (TX\_ACTIVE) features. The following links describe existing EM3x coexistence support features:
  - <http://community.silabs.com/t5/Mesh/WiFi-and-Zigbee-coordination/m-p/146392/highlight/true#M642>
  - <http://community.silabs.com/t5/Mesh-Knowledge-Base/EM3xx-Designing-for-radio-network-coexistence/ta-p/158864>

### 4.1 PTA Support Hardware Options

PTA is described in IEEE 802.15.2 (2003) Clause 6 and is a recommendation, not a standard. 802.15.2 originally addressed coexistence between 802.11b and 802.15.1 (Bluetooth Classic) and does not describe an exact hardware configuration. However, 802.15.2 recommends that the PTA implementation consider the following:

- TX REQUEST from 802.11b to PTA and TX REQUEST from 802.15.1 to PTA
- TX CONFIRM from PTA to 802.11b and TX CONFIRM from PTA to 802.15.1
- STATUS information from both radios:
  - Radio state [TX, RX, or idle]
  - Current and future TX/RX frequencies
  - Future expectation of a TX/RX start and duration
  - Packet type
  - Priority (Fixed, Randomized, or QoS based)

In considering radio state, transmit/receive, and frequencies, 802.15.2 describes the following.

Table 1. IEEE 802.15.2 2.4 GHz ISM Co-Located Radio Interference Possibilities

Co-located 802.11b State	Co-Located 802.15.1 State			
	Transmit		Receive	
	In-Band	Out-of-Band	In-Band	Out-of-Band
<b>Transmit</b>	Conflicting Transmits  Possible packet errors	No Conflict	Conflicting Transmit-Receive  Local packet received with errors	Conflicting Transmit-Receive  Local packet received with errors or no errors if sufficient isolation for frequency separation
<b>Receive</b>	Conflicting Transmit-Receive  Local packet received with errors	Conflicting Transmit-Receive  Local packet received with errors or no errors if sufficient isolation for frequency separation	Conflicting Receives  Possible packet errors	No Conflict

From the above table, the frequency separation recommendations from the section **Unmanaged Coexistence** remain required for managed coexistence:

- 802.15.2 “In-Band” is equivalent to Co-Channel (significant Wi-Fi impact on co-channel 802.15.4)
- 802.15.4 “Out-of-Band” covers both Adjacent and “Far-Away” Channel (~20 dB improvement in “Far-Away” Channel vs. Adjacent Channel)

As such, for Managed Coexistence, Silicon Labs recommends continuing to implement all of the Unmanaged Coexistence recommendations.

- Frequency Separation
- Operate Wi-Fi in 20 MHz Bandwidth
- Antenna Isolation
- Zigbee/Thread Retry Mechanisms
- FEM LNA in Bypass

In reviewing existing PTA implementations, Silicon Labs finds the PTA master implementation has been integrated into many Wi-Fi devices, but not all Wi-Fi devices support a PTA interface. The following figure shows the most common Wi-Fi/PTA implementations supporting Bluetooth.

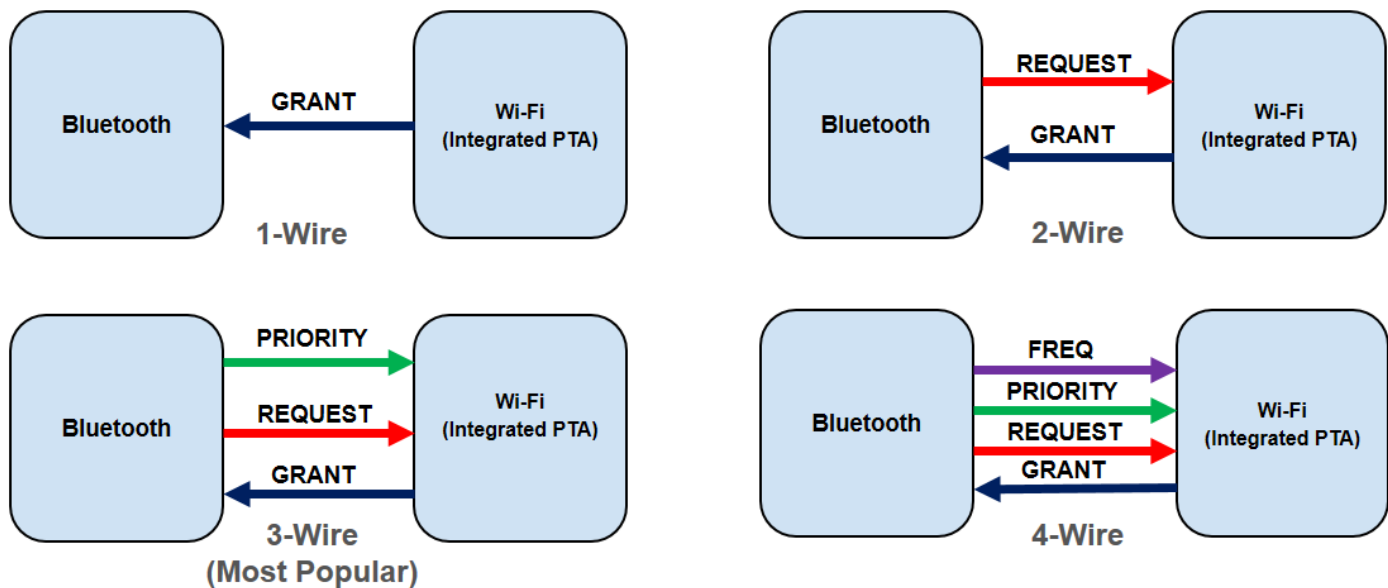


Figure 6. Typical Wi-Fi/Bluetooth PTA Implementations

#### 1-Wire PTA

In 1-Wire, the Wi-Fi/PTA device asserts a GRANT signal when Wi-Fi is not busy transmitting or receiving. When GRANT is asserted, the Bluetooth radio is allowed to transmit or receive. This mode does not allow external radio to request the 2.4 GHz ISM and is not recommended.

#### 2-Wire PTA

In 2-Wire, the REQUEST signal is added, allowing the Bluetooth radio to request the 2.4 GHz ISM band. The Wi-Fi/PTA device internally controls the prioritization between Bluetooth and Wi-Fi and, on a conflict, the PTA can choose to either GRANT Bluetooth or Wi-Fi.

#### 3-Wire PTA

In 3-Wire, the PRIORITY signal is added, allowing the Bluetooth radio to signify a high or low priority message is either being received or transmitted. The Wi-Fi/PTA device compares this external priority request against the internal Wi-Fi priority, which may be high/low or high/mid/low and can choose to either GRANT Bluetooth or Wi-Fi.

**Note:** PRIORITY can be implemented as static or time-shared (enhanced) priority. As of PTA support implementation in EmberZNet PRO 5.8.0 or later and Silicon Labs Thread 2.1.0 or later, Silicon Labs' EFR32 only supports static priority.

- Static: PRIORITY is either high or low during REQUEST asserted for the transmit or receive operation.
- Time-Shared: PRIORITY is either high or low for a typically 20  $\mu$ s duration after REQUEST asserted, but switches to low during receive operation and high during transmit operation.

Given the relatively low RF duty cycle of 802.15.4, static PRIORITY can be always asserted at the Wi-Fi/PTA input with the EFR32 PTA operating in 2-Wire mode. This frees a GPIO pin on the EFR32 and eliminates a circuit board trace.

#### 4-Wire PTA

In 4-Wire, the FREQ signal is added, allowing the Bluetooth radio to signify an "in-band" or "out-of-band" message is either being received or transmitted. Silicon Labs recommends maximizing frequency separation, making the FREQ signal mute. Silicon Labs' EFR32 does not support the FREQ signal and, for any 4-wire Wi-Fi/PTA with a FREQ input, Silicon Labs recommends asserting the FREQ input to the Wi-Fi/PTA.

##### 4.1.1 Single EFR32 Connected to Wi-Fi/PTA

Additional information about this topic is available in an expanded version of this application note, *AN1017-NDA: ZigBee and Thread Coexistence with Wi-Fi*, available under non-disclosure from Silicon Labs technical support.

## 4.1.2 Multiple EFR32s connected to Wi-Fi/PTA

Additional information about this topic is available in an expanded version of this application note, *AN1017-NDA: ZigBee and Thread Coexistence with Wi-Fi*, available under non-disclosure from Silicon Labs technical support.

### 4.1.3 Wi-Fi/PTA Considerations

Additional information about this topic is available in an expanded version of this application note, *AN1017-NDA: ZigBee and Thread Coexistence with Wi-Fi*, available under non-disclosure from Silicon Labs technical support.

## 4.2 PTA Support Software Setup

### 4.2.1 AppBuilder Configuration (PTA defaults after reset)

EmberZNet PRO 5.8.0 contains EFR32 PTA support, enabling customers to implement EFR32 PTA configured for target Wi-Fi/PTA platform. To enable EFR32 PTA coexistence support with 5.8.0:

1. Open AppBuilder from within Simplicity Studio V4.
2. Open the desired EFR32 application.
3. Select the **Plugins** tab.
4. Under “HAL” plugins, enable the “Coexistence Configuration” plugin, showing the following coexistence configuration options:

Name: Coexistence Configuration

Quality: Production ready

Description:

This plugin provides an interface for a customer to configure their coexistence GPIO interface. Customers should make sure that the GPIO pins chosen here do not conflict with any other GPIO used in their application. NOTE: This plugin is production quality for Zigbee but is currently alpha quality in Thread.

---

**Options:** [Reset to defaults](#)

☐ RHO(Radio Hold Off) signal enabled

☐ RHO(Radio Hold Off) active high

RHO(Radio Hold Off) signal GPIO port: C

RHO(Radio Hold Off) signal GPIO pin:[0-15] 11

☒ REQUEST signal enabled

☐ REQUEST signal is shared

☒ REQUEST signal active high

REQUEST signal GPIO port: C

REQUEST signal GPIO pin:[0-15] 10

REQUEST signal max backoff mask:[0-255] 15

☒ GRANT signal enabled

☐ GRANT signal active high

GRANT signal GPIO port: F

GRANT signal GPIO pin:[0-15] 3

☒ PRIORITY signal enabled

☒ PRIORITY signal active high

PRIORITY signal GPIO port: D

PRIORITY signal GPIO pin:[0-15] 12

☐ Receive retry REQUEST enabled

Receive retry timeout(milliseconds):[0-255] 16

☒ REQUEST high PRIORITY on receive retry

☐ Abort transmission mid packet if GRANT is lost

☒ TX high PRIORITY

☒ RX high PRIORITY

☒ Disable ACKING when GRANT deasserted, RHO asserted, or REQUEST not secured (shared REQUEST only)

Figure 7. Coexistence Configuration Plugin Settings

The coexistence configuration options are:

- **RHO (Radio Hold Off)**
  - **RHO (Radio Hold Off) signal enabled**
    - If selected, RHO is mapped to GPIO pin and is used by PTA implementation
    - If not selected, RHO is not mapped to GPIO pin and RHO is always deasserted
  - **RHO (Radio Hold Off) active high**
    - If selected, RHO is asserted when RHO GPIO pin is high ( $> V_{ih}$ )
    - If not selected, RHO is asserted when RHO GPIO pin is low ( $< V_{il}$ )
  - **RHO (Radio Hold Off) signal GPIO port and RHO (Radio Hold Off) signal GPIO pin**
    - Select RHO port and pin matching circuit board configuration
- **REQUEST**
  - **REQUEST signal enabled**
    - If selected, REQUEST is mapped to GPIO pin and is used by PTA implementation
    - If not selected, REQUEST is not mapped to GPIO pin
  - **REQUEST signal is shared**
    - If selected, REQUEST is shared and implements open-drain I/O for multi-EFR32 radio applications
    - Note: An external  $1\text{ k}\Omega \pm 5\%$  pull-up is required
    - If not selected, REQUEST is not shared and implements a push-pull output for single EFR32 radio applications
  - **REQUEST signal active high**
    - If selected, REQUEST GPIO pin is driven high ( $> V_{oh}$ ) when REQUEST is asserted
    - If not selected, REQUEST GPIO pin is driven low ( $< V_{ol}$ ) when REQUEST is asserted
  - **REQUEST signal GPIO port and REQUEST signal GPIO pin**
    - Select REQUEST port and pin matching circuit board configuration
  - **REQUEST signal max backoff mask[0-255]**
    - REQUEST signal max backoff determines the random REQUEST delay mask (only valid if REQUEST signal is shared)
    - Random delay (in  $\mu\text{s}$ ) is computed by masking the internal random variable against the entered mask
    - The mask should be set to a value of  $2^n - 1$  to insure a continuous random delay range
- **GRANT**
  - **GRANT signal enabled**
    - If selected, GRANT is mapped to GPIO pin and is used by PTA implementation
    - If not selected, GRANT is not mapped to GPIO pin and GRANT is always asserted
  - **GRANT signal active high**
    - If selected, GRANT is asserted when GRANT GPIO pin is high ( $> V_{ih}$ )
    - If not selected, GRANT is asserted when GRANT GPIO pin is low ( $< V_{il}$ )
  - **GRANT signal GPIO port and GRANT signal GPIO pin**
    - Select GRANT port and pin matching circuit board configuration
  - **Abort transmission mid packet if GRANT is lost**
    - If selected, losing GRANT during an 802.15.4 TX will abort the 802.15.4 TX
    - If not selected, losing GRANT after the initial evaluation at end of CCA will not abort the 802.15.4 TX
- **PRIORITY**
  - **PRIORITY signal enabled**
    - If selected, PRIORITY is mapped to GPIO pin and is used by PTA implementation
    - If not selected, PRIORITY is not mapped to GPIO pin
  - **PRIORITY signal active high**
    - If selected, PRIORITY GPIO pin is driven high ( $> V_{oh}$ ) when PRIORITY is asserted
    - If not selected, PRIORITY GPIO pin is driven low ( $< V_{ol}$ ) when PRIORITY is asserted
  - **PRIORITY signal GPIO port and PRIORITY signal GPIO pin**
    - Select REQUEST port and pin matching circuit board configuration
  - **TX high PRIORITY**
    - If selected, PRIORITY is asserted during 802.15.4 TX
    - If not selected, PRIORITY is deasserted during 802.15.4 TX
  - **RX high PRIORITY**
    - If selected, PRIORITY is asserted during 802.15.4 RX
    - If not selected, PRIORITY is deasserted during 802.15.4 RX

- **Receive Retry**

- **Receive retry REQUEST enabled**

- If selected, REQUEST is held after a corrupted receive packet until time-out expires or another packet is received

**Note:** This feature is useful to hold 2.4 GHz band clear while remote device re-transmits a packet, maximizing the opportunity to receive an uncorrupted retry packet from remote device, reducing 2.4 GHz RF traffic and improving battery life

- If not selected, REQUEST is not held after a corrupted receive packet

- **Receive retry timeout (milliseconds) [0-255]**

- Selects the timeout for REQUEST hold after a corrupted receive packet

**Note:** 16 ms is recommended to allow for maximum 802.15.4 packet duration and MAC retry random delay

**Note:** Many Wi-Fi/PTA implementations have a maximum GRANT timeout, which should be set to received retry timeout plus 6 ms to allow for maximum size corrupted packet, maximum random delay, and maximum size retry packet

- If not selected, PRIORITY GPIO pin is driven low (< Vol) when PRIORITY is asserted

- **REQUEST high PRIORITY on receive retry**

- If selected, PRIORITY is asserted during REQUEST hold after a corrupted receive packet
    - If not selected, PRIORITY is deasserted during REQUEST hold after a corrupted receive packet

- **Other**

- **Disable ACKing when GRANT deasserted, RHO asserted, or REQ not secured (shared REQUEST only)**

- If selected, the ACK to a valid RX packet, requiring an ACK, is not transmitted if GRANT is deasserted, RHO is asserted, or REQ is not secured (shared REQUEST only)

**Note:** This feature allows completing an 802.15.4 message, regardless of PTA signals, in order to minimize additional retries from remote device, reducing 2.4GHz RF traffic and improving battery life

- If not selected, the ACK to a valid RX packet requiring an ACK is transmitted regardless of GRANT, RHO, or REQ state

5. Complete other AppBuilder application setups and generate.

6. The coexistence configuration is saved in the application's .h file.

### Example 1: Configure EFR32 PTA support to operate as single EFR32 with typical time-slotted 3-Wire Wi-Fi/PTA

- Single EFR32 radio
- RHO unused
- REQUEST unshared, active high, PC10
  - Compatible 3-Wire Wi-Fi/PTA devices sometimes refer to this signal as RF\_ACTIVE or BT\_ACTIVE (active high)
- GRANT, active low, PF3
  - Compatible 3-Wire Wi-Fi/PTA devices sometimes refer to this signal as WLAN\_DENY (deny is active high, making grant active low)
- PRIORITY, active high, PD12
  - Compatible 3-Wire Wi-Fi/PTA devices sometimes refer to this signal as RF\_STATUS or BT\_STATUS (active high)
  - **Note:** PRIORITY is static, not time-shared. If operated with a 3-Wire Wi-Fi/PTA expecting time shared:
    - Static high PRIORITY is interpreted as high PRIORITY and always in TX mode, regardless of actual TX or RX
    - Static low PRIORITY is interpreted as low PRIORITY and always in RX mode, regardless of actual TX or RX
- Other options enabled to maximize 802.15.4 performance:
  - 802.15.4 RX and TX both at high priority
  - Receive retry REQUEST enabled with 16ms time-out and high priority
  - Enabled ACKing when GRANT deasserted

Name: Coexistence Configuration

Quality: Production ready

Description:

This plugin provides an interface for a customer to configure their coexistence GPIO interface. Customers should make sure that the GPIO pins chosen here do not conflict with any other GPIO used in their application. NOTE: This plugin is production quality for Zigbee but is currently alpha quality in Thread.

**Options:** [Reset to defaults](#)

☐ RHO(Radio Hold Off) signal enabled

☐ RHO(Radio Hold Off) active high

RHO(Radio Hold Off) signal GPIO port: C

RHO(Radio Hold Off) signal GPIO pin:[0-15] 11

☒ REQUEST signal enabled

☐ REQUEST signal is shared

☒ REQUEST signal active high

REQUEST signal GPIO port: C

REQUEST signal GPIO pin:[0-15] 10

REQUEST signal max backoff mask:[0-255] 15

☒ GRANT signal enabled

☐ GRANT signal active high

GRANT signal GPIO port: F

GRANT signal GPIO pin:[0-15] 3

☒ PRIORITY signal enabled

☒ PRIORITY signal active high

PRIORITY signal GPIO port: D

PRIORITY signal GPIO pin:[0-15] 12

☒ Receive retry REQUEST enabled

Receive retry timeout(milliseconds):[0-255] 16

☒ REQUEST high PRIORITY on receive retry

☐ Abort transmission mid packet if GRANT is lost

☒ TX high PRIORITY

☒ RX high PRIORITY

☐ Disable ACKing when GRANT deasserted, RHO asserted, or REQUEST not secured (shared REQUEST only)

**Figure 8. EFR32 PTA Support Configured to Operate as Single EFR32 with Typical Time-Slotted 3-Wire Wi-Fi/PTA**



The following logic analyzer capture shows the PTA interface, WiFi radio state, and EFR32 radio state for an EFR32 radio configured for time-slotted 3-Wire Wi-Fi/PTA:

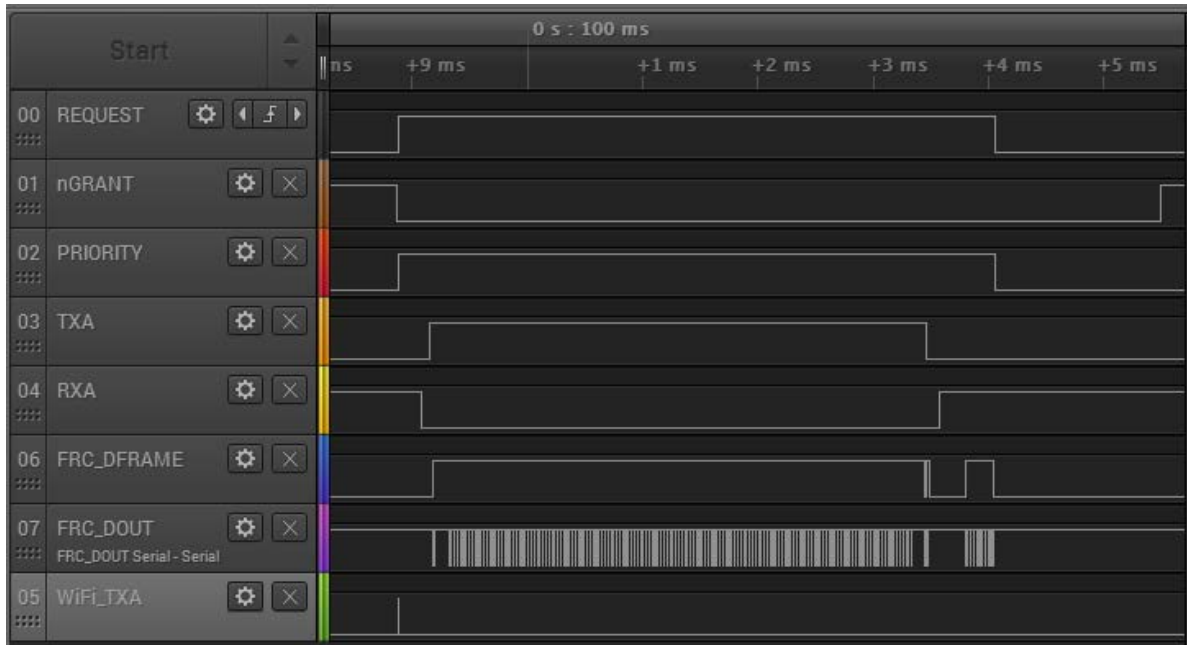


Figure 9. Example 802.15.4 TX for Single EFR32 time-slotted 3-Wire Wi-Fi/PTA Logic Analyzer Capture

Where:

**REQUEST:** active high, push-pull REQUEST output  
**nGRANT:** active low GRANT input  
**PRIORITY:** active high PRIORITY output  
**TXA:** EFR32 FEM TX Active control signal (configured via FEM Control plugin)  
**RXA:** EFR32 FEM RX Active control signal (configured via FEM Control plugin)  
**FRC\_DFRAME:** EFR32 Frame Control Data Frame signal (packet trace frame/synch)  
**FRC\_DOUT:** EFR32 Frame Control Data Out signal (packet trace data)  
**WiFi\_TXA:** Wi-Fi TX Active signal


This logic analyzer sequence shows:

1. Wi-Fi starts a transmit, but is immediately pre-empted (WiFi\_TXA pulse) by higher priority 802.15.4 transmit asserting REQUEST and PRIORITY.
2. GRANT is asserted by Wi-Fi/PTA.
3. EFR32 radio completes CCA and CCA passes and GRANT is asserted.
4. EFR32 radio proceeds with transmit (RXA deasserts, followed by TXA assert).
5. After transmit, EFR32 waits for ACK (TXA deasserts, followed by RXA assert).
6. EFR32 receives ACK (second FRC\_DFRAME pulse). *<= 802.15.4 TX message successfully completed*
7. EFR32 deasserts PRIORITY and REQUEST.
8. Wi-Fi/PTA deasserts GRANT.

## Example 2: Configure EFR32 PTA support to operate with multi-radio 2-Wire PTA

- Multiple EFR32 radios (external 1 kΩ ±5% pull-up required)
- RHO unused
- REQUEST shared, active low, PC10
- GRANT, active low, PF3
- PRIORITY unused
- Other options enabled to maximize 802.15.4 performance:
  - Receive retry REQUEST enabled with 16 ms time-out
  - Enabled ACKing when GRANT deasserted

Name: Coexistence Configuration

Quality:  Production ready

Description:

This plugin provides an interface for a customer to configure their coexistence GPIO interface. Customers should make sure that the GPIO pins chosen here do not conflict with any other GPIO used in their application. NOTE: This plugin is production quality for Zigbee but is currently alpha quality in Thread.

### Options

[Reset to defaults](#)

☐ RHO(Radio Hold Off) signal enabled

☐ RHO(Radio Hold Off) active high

RHO(Radio Hold Off) signal GPIO port: C

RHO(Radio Hold Off) signal GPIO pin:[0-15] 11

☒ REQUEST signal enabled

☒ REQUEST signal is shared

☐ REQUEST signal active high

REQUEST signal GPIO port: C

REQUEST signal GPIO pin:[0-15] 10

REQUEST signal max backoff mask:[0-255] 15

☒ GRANT signal enabled

☐ GRANT signal active high

GRANT signal GPIO port: F

GRANT signal GPIO pin:[0-15] 3

☐ PRIORITY signal enabled

☒ PRIORITY signal active high

PRIORITY signal GPIO port: D

PRIORITY signal GPIO pin:[0-15] 12

☒ Receive retry REQUEST enabled

Receive retry timeout(milliseconds):[0-255] 16

☒ REQUEST high PRIORITY on receive retry

☐ Abort transmission mid packet if GRANT is lost

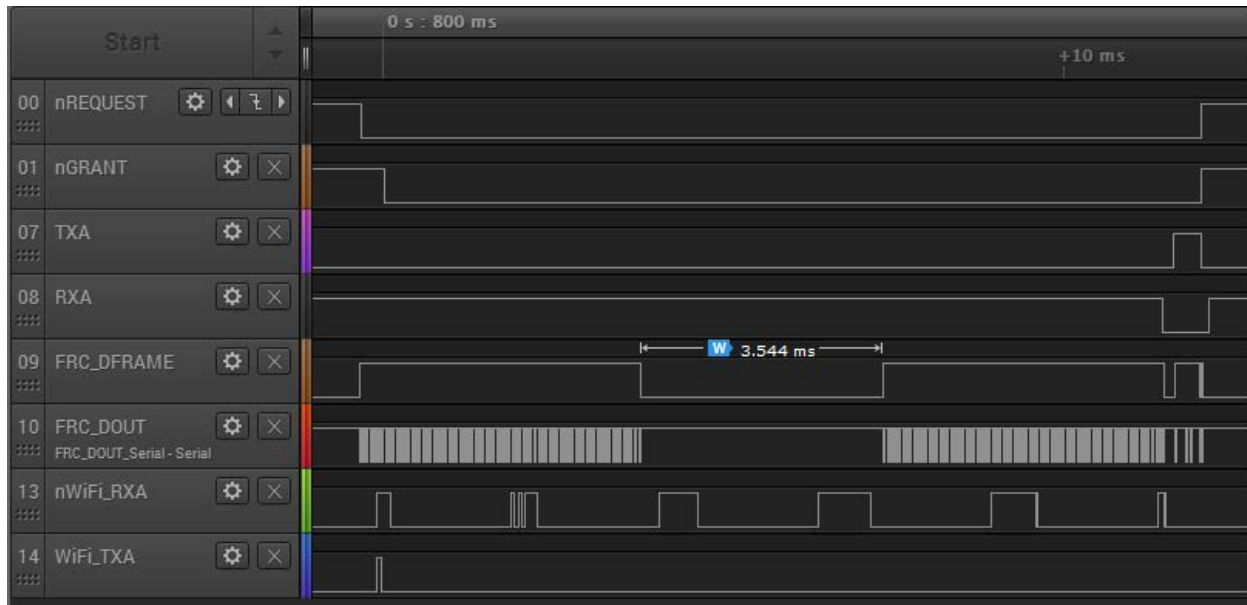
☒ TX high PRIORITY

☒ RX high PRIORITY

☐ Disable ACKing when GRANT deasserted, RHO asserted, or REQUEST not secured (shared REQUEST only)

Figure 10. EFR32 PTA Support Configures to operate with Multi-radio 2-Wire PTA

The following logic analyzer capture shows the PTA interface, WiFi radio state, and EFR32 radio state for an EFR32 radio configured for multi-radio 2-Wire PTA:



**Figure 11. Example 802.15.4 RX for Multi-EFR32 2-Wire Wi-Fi/PTA Logic Analyzer Capture**

Where:

- nREQUEST:** active low, shared (open-drain) REQUEST input/output
- nGRANT:** active low GRANT input
- TXA:** EFR32 FEM TX Active control signal (configured via FEM Control plugin)
- RXA:** EFR32 FEM RX Active control signal (configured via FEM Control plugin)
- FRC\_DFRAME:** EFR32 Frame Control Data Frame signal (packet trace frame/synch)
- FRC\_DOUT:** EFR32 Frame Control Data Out signal (packet trace data)
- nWiFi\_RXA:** Wi-Fi RX Active signal
- WiFi\_TXA:** Wi-Fi TX Active signal

This logic analyzer sequence shows:

1. 802.15.4 packet is detected (FRC\_DFRAME asserted) while Wi-Fi is receiving a packet (nWiFi\_RXA asserted).
2. Shared REQUEST signal is tested and found not asserted by another EFR32 radio, so receiving EFR32 radio asserts REQUEST.
3. Wi-Fi ACK is transmitted (WiFi\_TXA asserted) during 802.15.4 receive (no Wi-Fi TX pre-emption or higher priority Wi-Fi activity).
4. After Wi-Fi ACK completes, GRANT is asserted by Wi-Fi/PTA.
5. 802.15.4 receive is completed but CRC failed as packet was corrupted by co-located Wi-Fi ACK transmit during receive.
6. Since PTA configured with *Receive retry REQUEST enabled* using 16 ms time-out, REQUEST is held up to 16 ms for 802.15.4 retry with 2.4 GHz quiet (Wi-Fi held off).
7. Wi-Fi continues to receive packets (nWiFi\_RXA asserts), but does not ACK while 802.15.4 radio has GRANT.
8. After 3.5 ms gap for end-node ACK time-out and MAC random delay, the 802.15.4 retry packet arrives and is received without error.
9. 802.15 ACK is transmitted (TXA asserted). *<= 802.15.4 RX message successfully completed*
10. After 802.15.4 ACK completes, REQUEST is deasserted, followed by GRANT deassert.

## 4.2.2 Run-Time PTA Re-configuration

The following PTA options can be re-configured at run-time:

- Enable or disable PTA
- Receive retry timeout (milliseconds) [0-255]
- Disable ACKing when GRANT deasserted, RHO asserted, or REQ not secured (shared REQUEST only)
- Abort transmission mid packet if GRANT is lost
- TX high PRIORITY
- RX high PRIORITY
- REQUEST high PRIORITY on receive retry
- Receive retry REQUEST enabled
- RHO(Radio Hold Off) signal enabled

The API function calls for re-configuring vary based on SoC, EZSP, or TMSP application.

### 4.2.2.1 SoC Application

The following two SoC API function calls enable and disable the PTA at run-time:

```
bool halPtaIsEnabled(void);
EmberStatus halPtaSetEnable(bool enabled);
```

The following two SoC API function calls re-configure the PTA at run-time:

```
HalPtaOptions halPtaGetOptions(void);
EmberStatus halPtaSetOptions(HalPtaOptions options);
```

Where HalPtaOptions is an uint32\_t with the following bit-map definition:

PTA Feature	Bit Position	Size (bits)
Receive retry timeout (milliseconds) [0-255]	0	8
Disable ACKing when GRANT deasserted, RHO asserted, or REQ not secured (shared REQUEST only)	8	1
Abort transmission mid packet if GRANT is lost	9	1
TX high PRIORITY	10	1
RX high PRIORITY	11	1
REQUEST high PRIORITY on receive retry	12	1
Receive retry REQUEST enabled	13	1
RHO(Radio Hold Off) signal enabled	14	1
Reserved (Reserved bits MUST be written 0)	15	17

### 4.2.2.2 Zigbee Network Coprocessor Application using EZSP API

The following two EZSP (EmberZNet Serial Protocol) API function calls enable and disable the PTA and re-configure the PTA at run-time:

```
EzspStatus ezspGetValue(EzspValueId valueId, uint8_t *valueLength, uint8_t *value);
EzspStatus ezspSetValue(EzspValueId valueId, uint8_t valueLength, uint8_t *value);
```

Where valueId and valueLength have the following PTA related options:

EZSP Value ID	Value	Length (bytes)	Description
EZSP_VALUE_ENABLE_PTA	0x31	1	Enable (1) or disable (0) packet traffic arbitration.
EZSP_VALUE_PTA_OPTIONS	0x32	4	Set packet traffic arbitration configuration options.

Where PTA configuration options are an uint32\_t with the following bit-map definition:

PTA Feature	Bit Position	Size (bits)
Receive retry timeout (milliseconds) [0-255]	0	8
Disable ACKing when GRANT deasserted, RHO asserted, or REQ not secured (shared REQUEST only)	8	1
Abort transmission mid packet if GRANT is lost	9	1
TX high PRIORITY	10	1
RX high PRIORITY	11	1
REQUEST high PRIORITY on receive retry	12	1
Receive retry REQUEST enabled	13	1
RHO(Radio Hold Off) signal enabled	14	1
Reserved (Reserved bits MUST be written 0)	15	17

#### 4.2.2.3 Thread Network Coprocessor Application using TMSP API

The following two TMSP (Thread Management Serial Protocol) API function calls enable and disable the PTA at run-time:

```
bool halPtaIsEnabled(void);
EmberStatus halPtaSetEnable(bool enabled);
```

The following two TMSP API function calls re-configure the PTA at run-time:

```
HalPtaOptions halPtaGetOptions(void);
EmberStatus halPtaSetOptions(HalPtaOptions options);
```

Where HalPtaOptions is an uint32\_t with the following bit-map definition:

PTA Feature	Bit Position	Size (bits)
Receive retry timeout (milliseconds) [0-255]	0	8
Disable ACKing when GRANT deasserted, RHO asserted, or REQ not secured (shared REQUEST only)	8	1
Abort transmission mid packet if GRANT is lost	9	1
TX high PRIORITY	10	1
RX high PRIORITY	11	1
REQUEST high PRIORITY on receive retry	12	1
Receive retry REQUEST enabled	13	1
RHO(Radio Hold Off) signal enabled	14	1
Reserved (Reserved bits MUST be written 0)	15	17

### 4.3 Test Results

Additional information about this topic is available in an expanded version of this application note, *AN1017-NDA: ZigBee and Thread Coexistence with Wi-Fi*, available under non-disclosure from Silicon Labs technical support.

## 5 Additional PTA Backplane Evaluation Board (EVB)

Silicon Labs' EFR32 PTA solution can be evaluated by ordering an EFR32™ Mighty Gecko Wireless SoC Starter Kit (WSTK) #SLWSTK6000A, and requesting a PTA Backplane EVB (#WBFA-0001) from Silicon Labs' Sales. The PTA Backplane supports using one or more of the three EFR32MG EVBs in the wireless mesh development kit and provides easy PTA signal connections to Wi-Fi/PTA EVBs via either +3.3 V or +1.8 V I/O.

When using the PTA Backplane EVB for PTA evaluation, the AppBuilder coexistence-configuration must have any enabled PTA signals connected to the following GPIO:

EFR32 PTA Signal	EFR32 GPIO
RHO	PC11
REQUEST	PC10
GRANT	PF3
PRIORITY	PD12

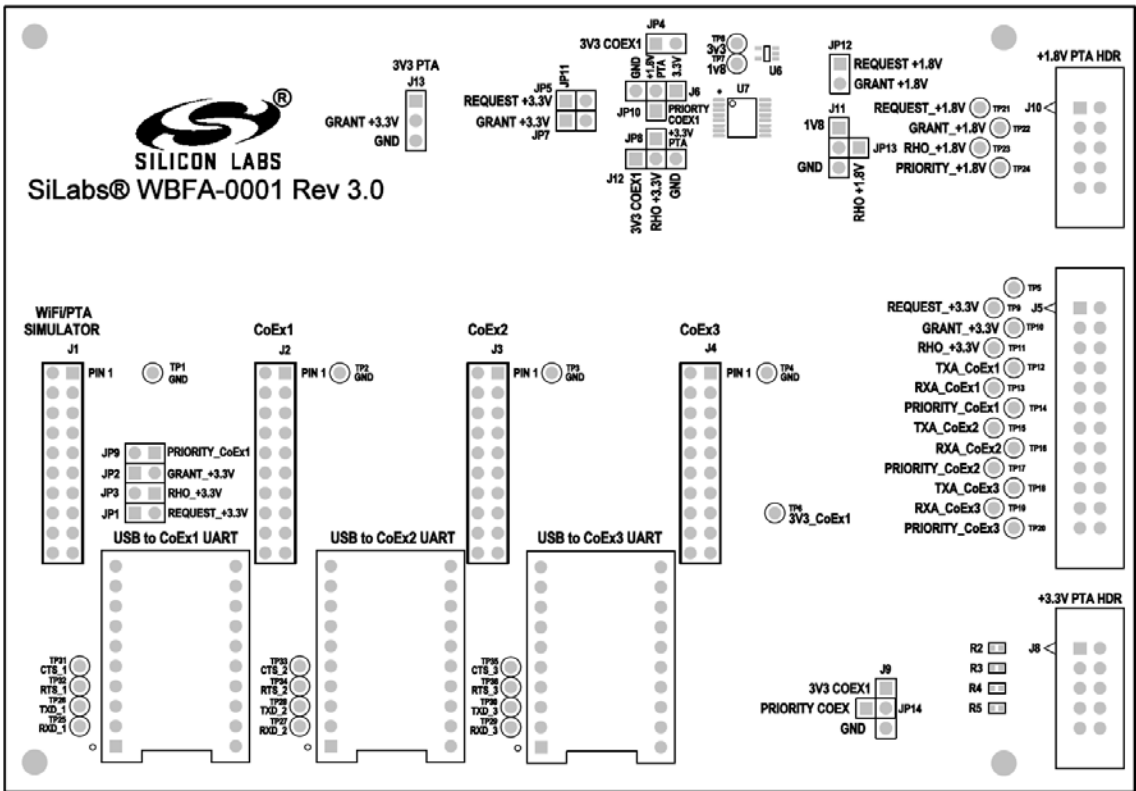
It is also helpful to observe the FEM control signals by using the FEM Control plugin and directing TXA and RXA as follows:

FEM Control Signals	EFR32 PRS Channel	EFR32 PRS Location	EFR32 GPIO
TXA	5	0	PD10
RXA	6	13	PD11

It is also helpful to observe the FRC DFRAME and FRC DOUT packet trace signals on the WSTK J101 header:

FRC Signal	WSTK J101
FRC_DFRAME	P22
FRC_DOUT	P24

The PTA Backplane EVB has the connector and jumper options shown below:



### Figure 12. PTA Backplane EVB Connector and Jumper Options

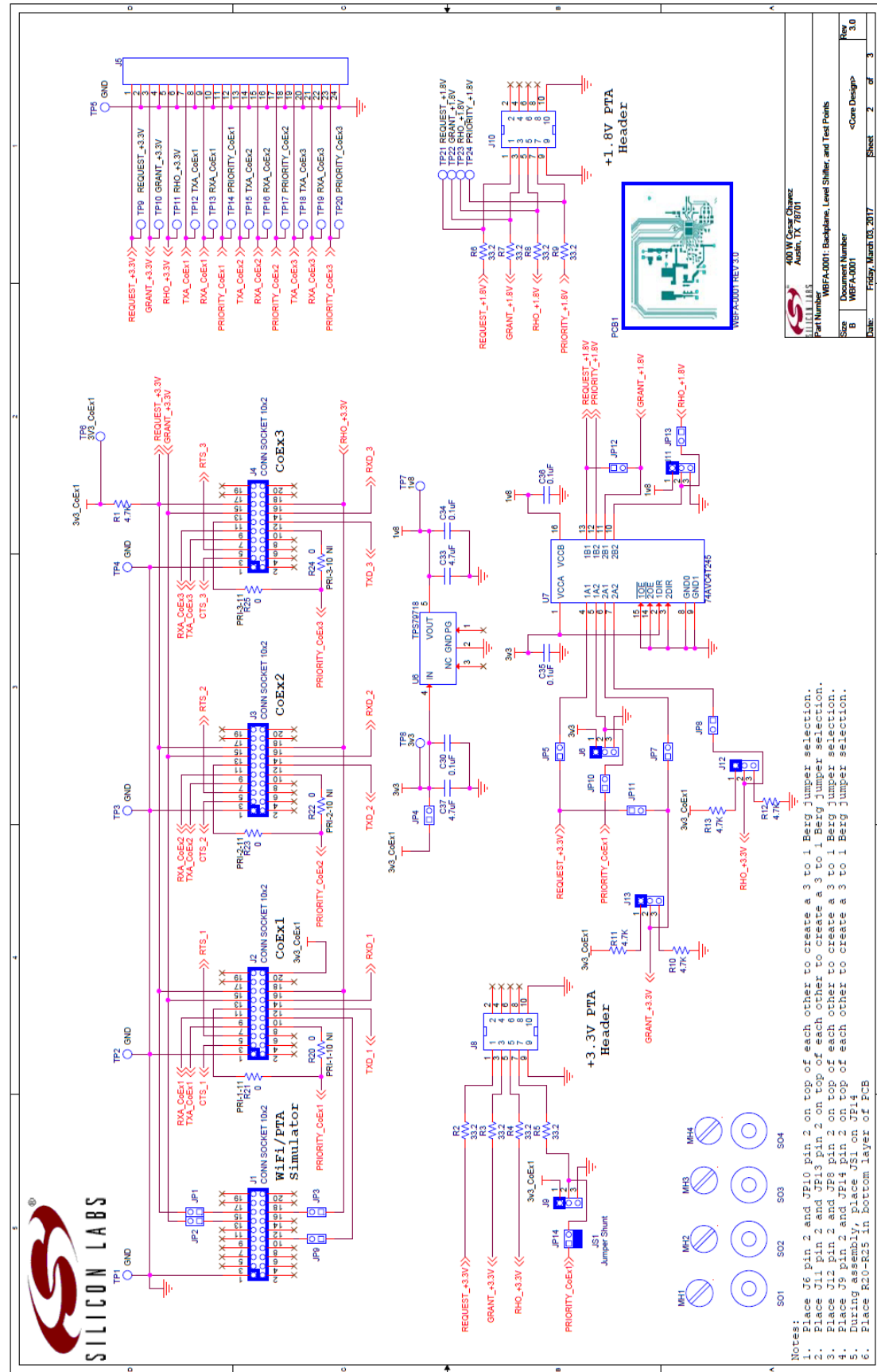


Figure 13. WBFA-0001 Schematic (1/2)



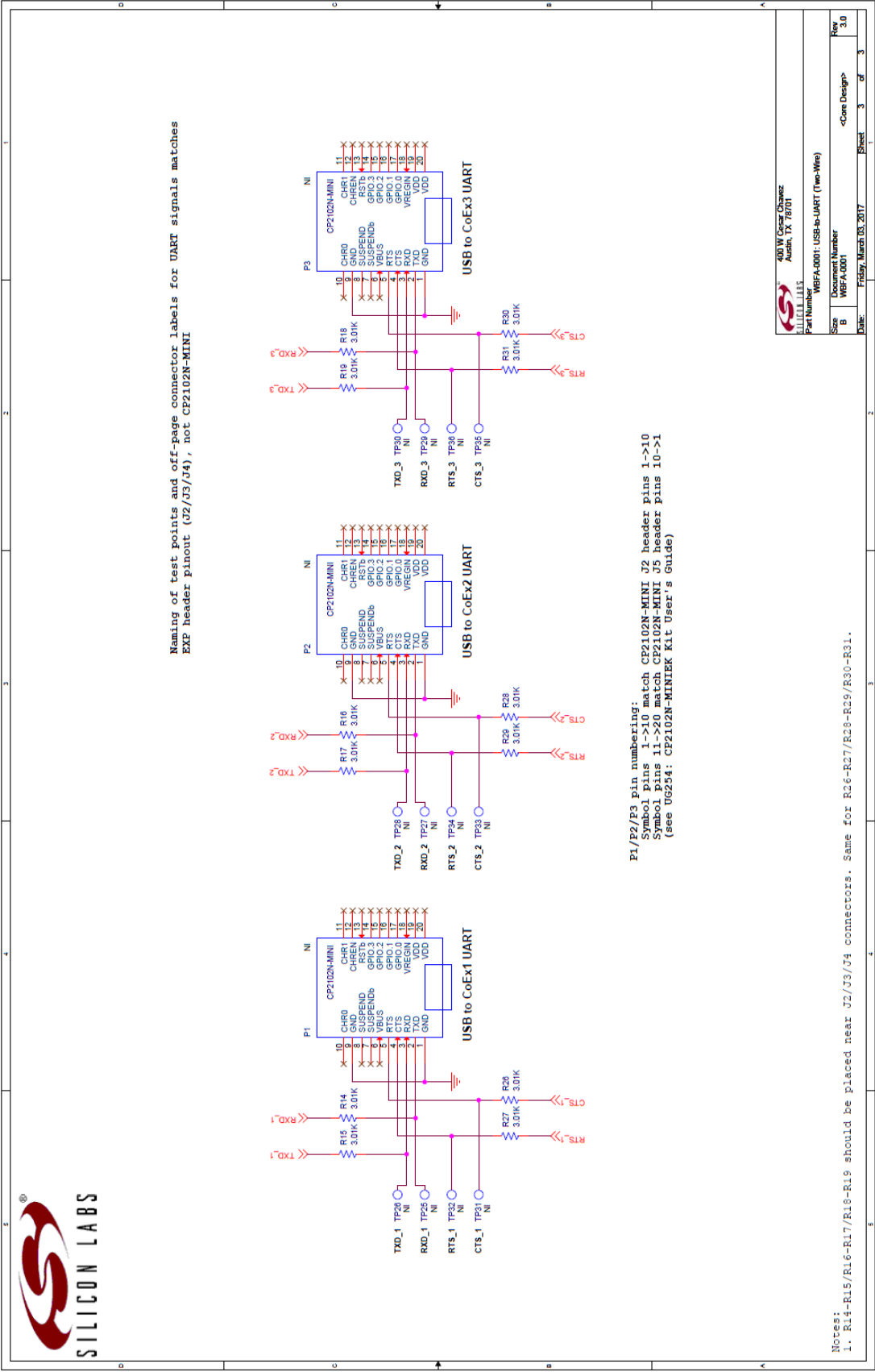


Figure 33. WBFA-0001 Schematic (2/2)

## 5.1 +3.3 V or +1.8 V I/O to Wi-Fi/PTA Device

The PTA Backplane EVB provides the capability of interfacing to Wi-Fi/PTA devices via +3.3V or +1.8V I/O as needed.

### 5.1.1 +3.3 V I/O

For +3.3 V I/O to Wi-Fi/PTA device, if PRIORITY (CoEx1 slot only) is not used, no jumpers are necessary. If PRIORITY (CoEx1 slot only) is used, populate the JP14 (blue) jumper as shown below:

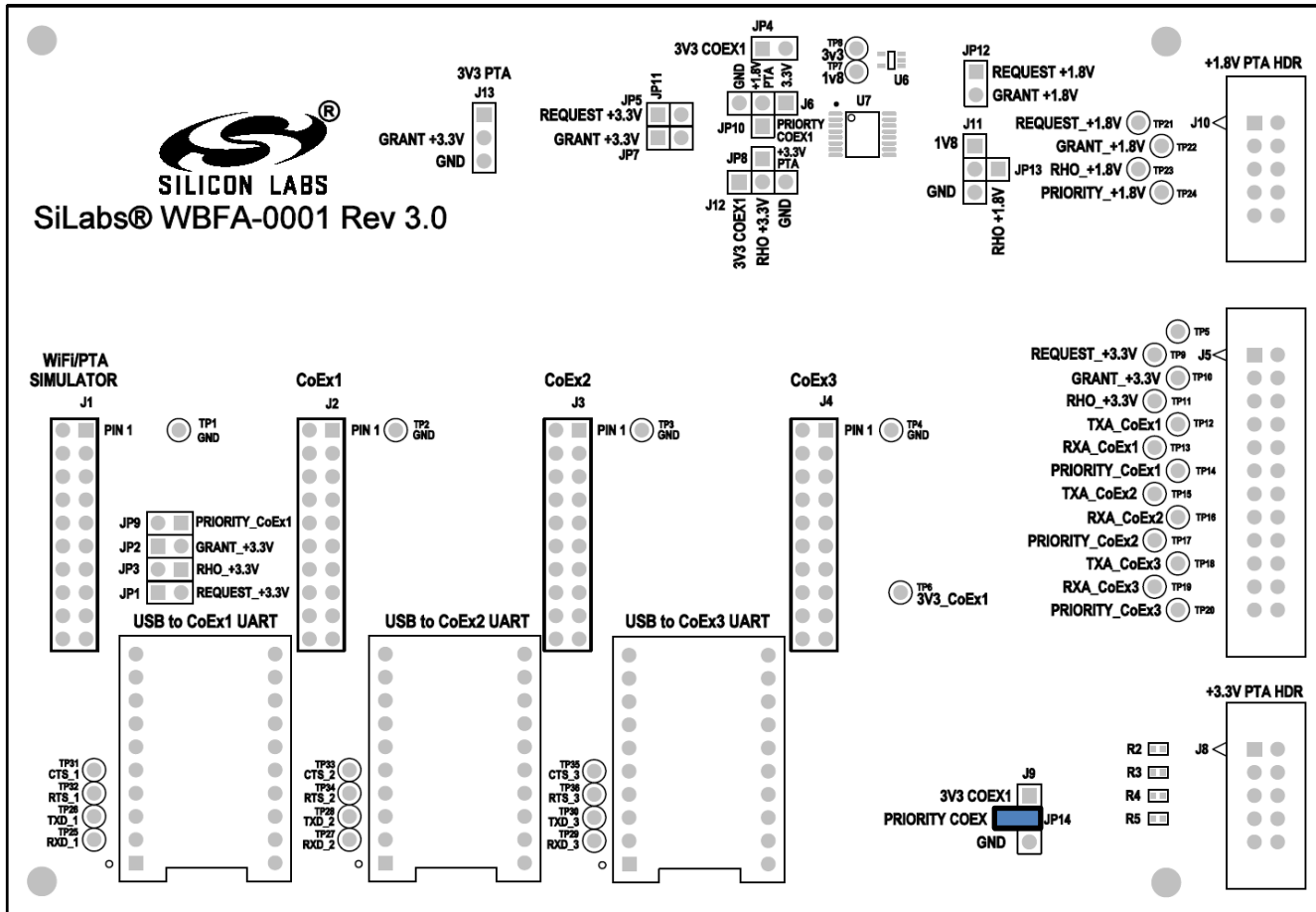


Figure 34. PTA Backplane EVB Jumpers for +3.3V I/O Operation

The +3.3V /IO connections to Wi-Fi/PTA device can be made via +3.3V PTA HDR (J8) as follows:

EFR32 PTA Signal	J8 Pin
REQUEST_+3.3V	1
GRANT_+3.3V	3
RHO_+3.3V	5
PRIORITY_CoEx1 (+3.3V)	7
GND	9 and 10

## 5.1.2 +1.8 V I/O

For +1.8 V I/O to Wi-Fi/PTA device, populate all red jumper shown below. If PRIORITY (CoEx1 slot only) is not used, no additional jumpers are necessary. If PRIORITY (CoEx1 slot only) is used, populate the JP6 (blue) jumper as shown below:

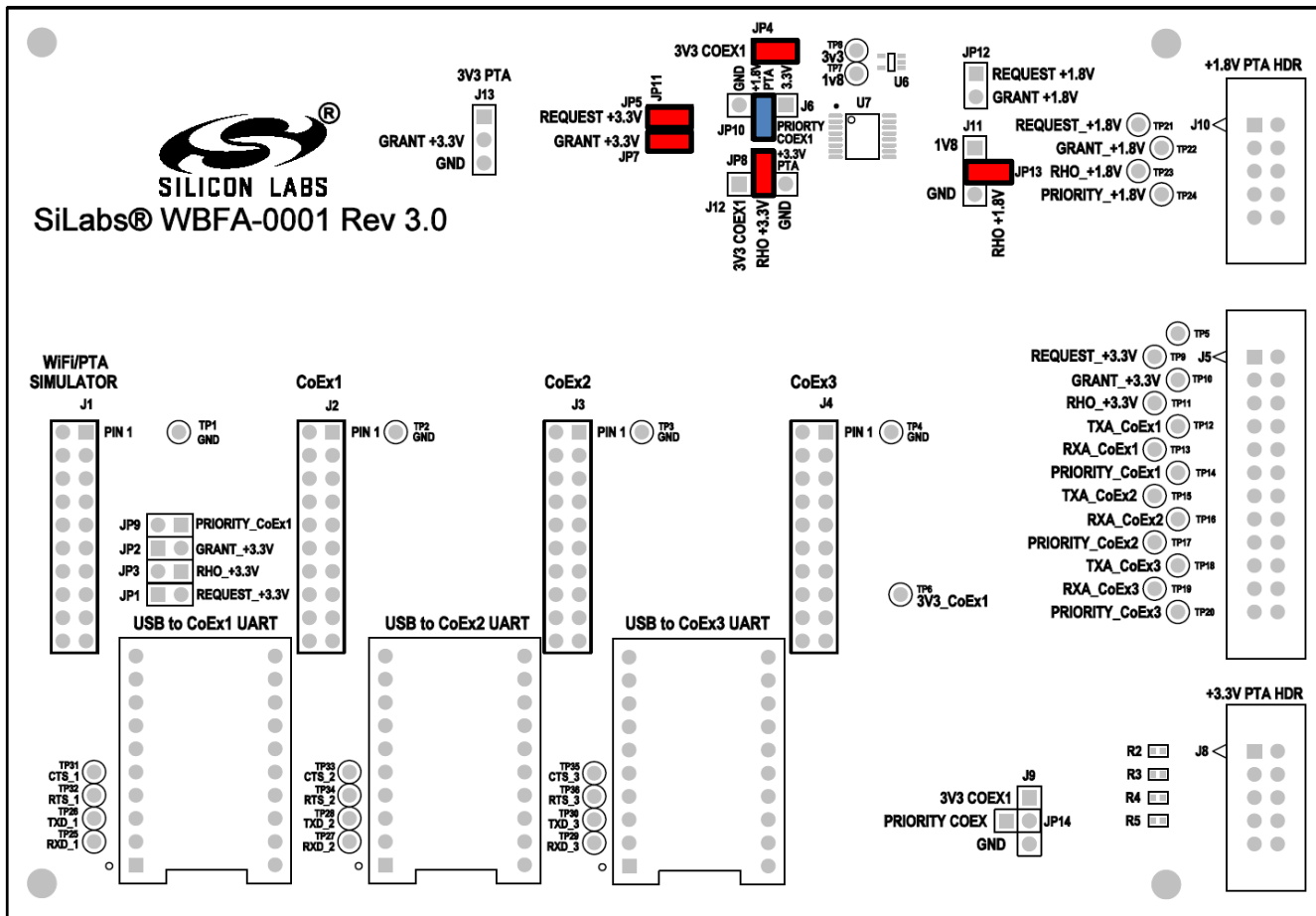


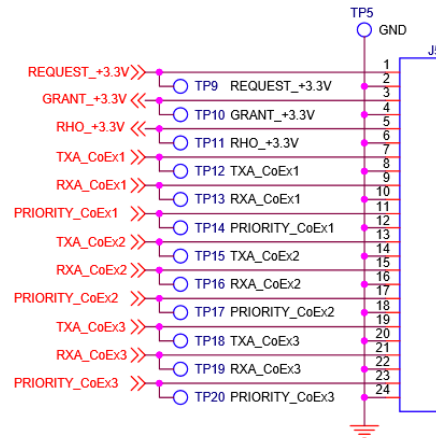
Figure 35. PTA Backplane EVB Jumpers for +1.8V I/O Operation

The +3.3 V I/O connections to Wi-Fi/PTA device can be made via +1.8 V PTA HDR (J10) as follows:

EFR32 PTA Signal	J10 Pin
REQUEST_+1.8V	1
GRANT_+1.8V	3
RHO_+1.8V	5
PRIORITY_CoEx1 (+1.8V)	7
GND	9 and 10

## 5.2 Logic Analyzer Observation

The EFR32 PTA signals (+3.3V I/O) are observable via a logic analyzer connected to J5 as shown below:



**Figure 36. PTA Backplane EVB Header for PTA Signal Logic Analyzer Connection**

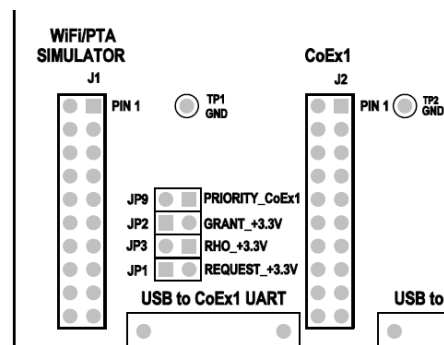
Where:

- CoEx1 refers to EFR32/WSTK connected into CoEx1 header (J2)
- CoEx2 refers to EFR32/WSTK connected into CoEx2 header (J3)
- CoEx3 refers to EFR32/WSTK connected into CoEx1 header (J4)

## 5.3 Single-EFR32 Radio

For a single EFR32 radio, be sure to:

1. Develop the desired PTA test application using AppBuilder.
2. Add the coexistence-configuration plugin and configure for target Wi-Fi/PTA device as per single-radio description in section **PTA Support Software Setup**.
3. Add FEM Control plugin (configured as described above) to enable TXA and RXA observation.
4. Build the EFR32 application and program the WSTK.
5. Plug the EXP header on the EFR32/WSTK EVB into PTA Backplane EVB CoEx1 header (J2) (ensure pin 1 to pin 1) as shown in the following figure.



**Figure 37. PTA Backplane EVB Header for Single EFR32 Radio Testing**

1. Enable PTA on the Wi-Fi/PTA device.
2. Execute Wi-Fi stream and 802.15.4 stream.
3. Observe Wi-Fi error rate and 802.15.4 message success.
4. Observe PTA signals to debug, and adjust as necessary.
5. With the PTA solution confirmed, the coexistence-plugin only needs to be modified for the PTA GPIOs assigned on target hardware.

## 5.4 Multi-EFR32 Radio

For multi- EFR32 radio, be sure to:

1. Develop desired PTA test applications using AppBuilder.
2. Add coexistence-configuration plugins and configure for target Wi-Fi/PTA device as per multi-radio description in section **PTA Support Software Setup**.
3. Add the FEM Control plugin (configured as described above) to enable TXA and RXA observation.
4. Build the EFR32 applications and program the WSTKs.
5. Plug the EXP header on one EFR32/WSTK EVB into PTA Backplane EVB CoEx1 header (J2) and the second and third EFR32/WSTK EVBs, if used, into CoEx2 and CoEx3 headers (J3 and/or J4) (ensure pin 1 to pin 1) as shown in the following figure.

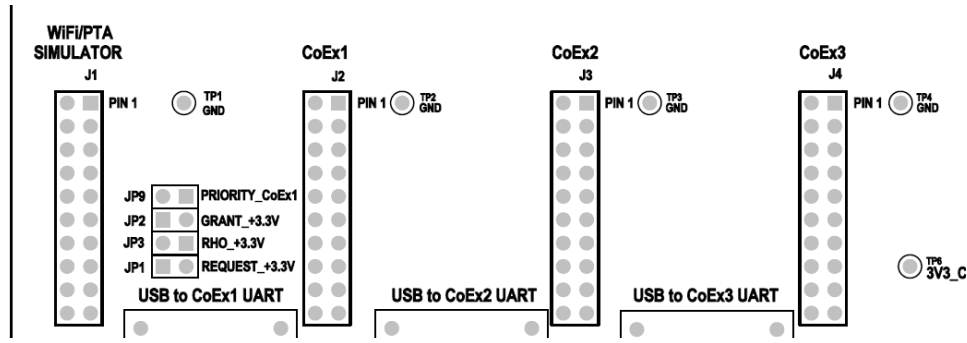


Figure 38. PTA Backplane EVB Headers for Multi-EFR32 Radio Testing

1. Enable PTA on the Wi-Fi/PTA device.
2. Execute Wi-Fi stream and 802.15.4 streams.
3. Observe Wi-Fi error rate and 802.15.4 message success.
4. Observe PTA signals to debug and adjust as necessary.
5. With the PTA solution confirmed, the coexistence-plugin only needs to be modified for the PTA GPIOs assigned on target hardware.

## 5.5 Testing xNCP Applications

If testing xNCP (customized NCP) images with PTA support, initial testing may benefit from enabling the EFR32 serial connection through the WSTK EXP header, connecting to a UART-to-USB adapter, and testing using a PC-based gateway host application to drive the EFR32's xNCP image. This can be accomplished through the following procedure:

1. Order a CP2102N-MINIEK for each EFR32 xNCP on the PTA Backplane EVB.

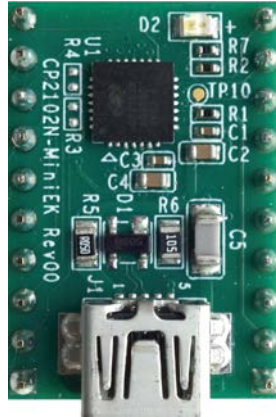
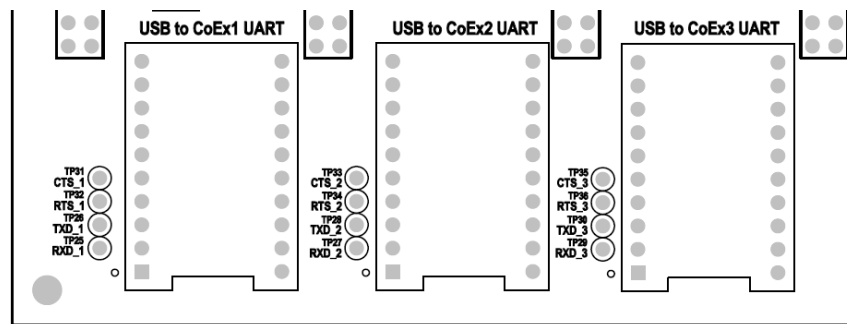


Figure 39. CP2102N-MINIEK Pin-Out

2. Install the CP2102 drivers onto the test PC.
3. Solder each CP2102N-MINIEK to PTA Backplane EVB's CoEx1 (P1), CoEx2 (P2), and/or CoEx3 (P3) as follows:



**Note:** Each CP2102N-MINIEK is connected to its respective EFR32/WSTK EVB as follows:

CP2102N-MINIEK Signal	J2, J3, or J4 header
GND	Pin 1 (GND)
TXD	Pin 14 (UART_RX, PA1)
RXD	Pin 12 (UART_TX, PA0)
CTS	Pin 5 (UART_RTS, PA3)
RTS	Pin 3 (UART_CTS, PA2)

4. Within AppBuilder, under **Core plugins** enable the NCP-UART plugin and set **Flow Control Type** to desired (**Hardware** or **Software**).
5. Enable and configure the Coexistence-Configuration and FEM Control plugins.
6. On the **Other** tab, under "**Additional Macros**", ensure that the **EZSP\_UART** and **NO\_USB** macros are defined and selected.
7. After generating and compiling the xNCP application, program the EFR32.
8. Install the EFR32/WSTK into the PTA Backplane EVB, then connect associated the CP2102N-MINIEK to the PC.
9. Note the COM port assigned to CP2102N-MINIEK.
10. Start the PC-based gateway application as appropriate for the desired flow control:
  - Hardware flow control at 115.2Kb: using `-n 0 -p COMx` arguments, where x is the assigned CP2102N-MINIEK COM port.
  - Software flow control at 57.6 Kb: using `-n 1 -p COMx` arguments, where x is the assigned CP2102N-MINIEK COM port.
11. Test the PTA solution using the PC-gateway application.

## 6 Conclusions

Co-located, strong Wi-Fi can have a substantial impact on 802.15.4 performance. 802.15.4 performance with co-located Wi-Fi can be improved through unmanaged and managed coexistence techniques. Unmanaged coexistence recommendations include:

1. Implement frequency separation.
2. Operate Wi-Fi with 20.MHz bandwidth.
3. Increase antenna isolation.
4. Use zigbee/Thread Retry Mechanisms.
5. Remove FEM (or operate FEM LNA in bypass).

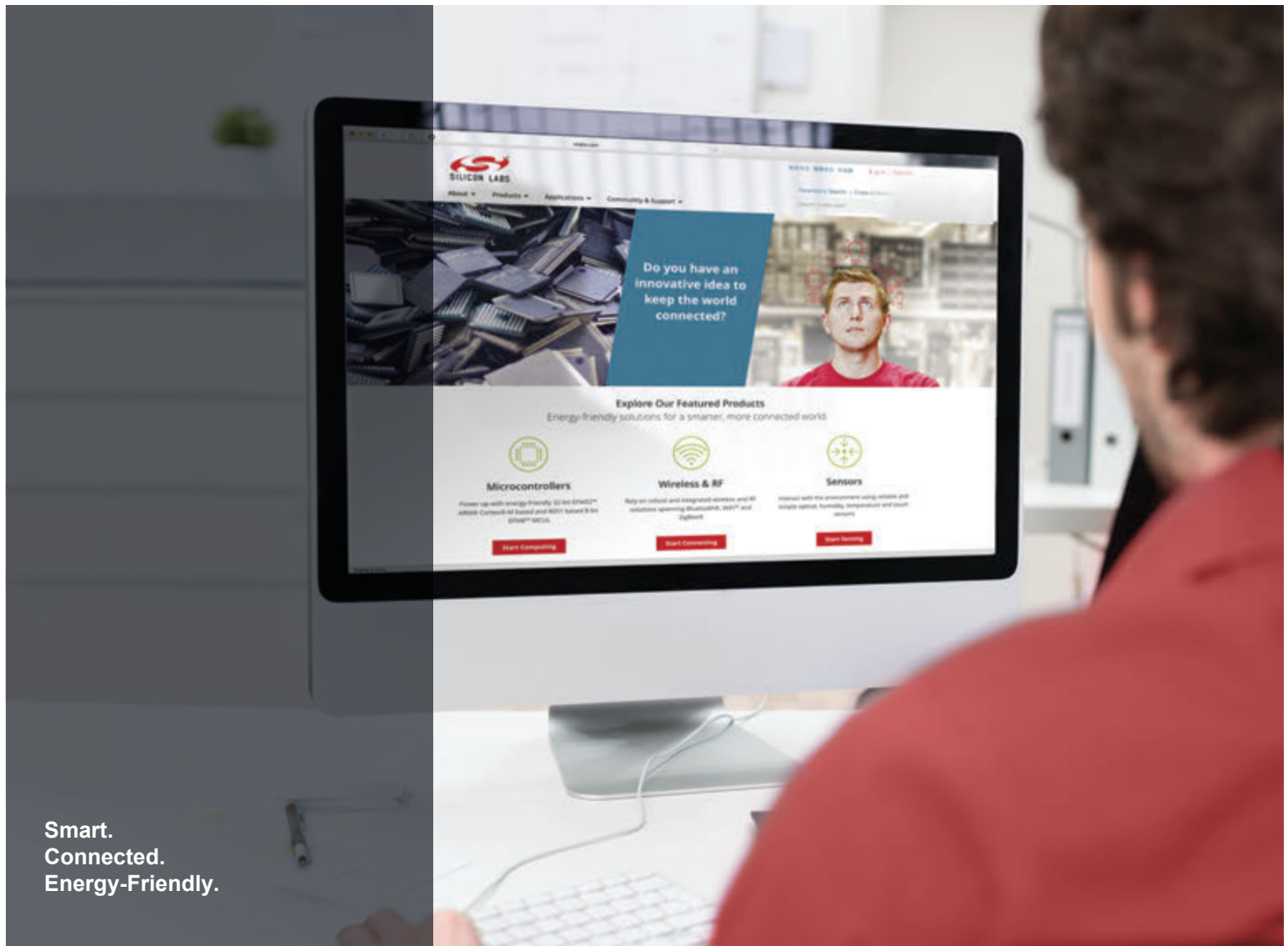
With market trends toward higher Wi-Fi TX power, higher Wi-Fi throughput, and integration of Wi-Fi and 802.15.4 radios into the same device, unmanaged techniques alone may prove insufficient, so that a managed coexistence solution is required. Even with a managed coexistence solution, all unmanaged coexistence recommendations are still necessary. Managed coexistence utilizes:

1. Wi-Fi/PTA devices providing 802.15.2-derived Packet Traffic Arbitration.
2. Silicon Labs' EFR32 PTA solution:
  1. One to four GPIOs implementing a combination of REQUEST, GRANT, PRIORITY, and RHO.
  2. Silicon Labs' AppBuilder coexistence-configuration plugin to configure EFR32 PTA support for available GPIO pins and for compatibility with the chosen Wi-Fi/PTA device.
  3. Silicon Labs' API, supporting run-time PTA reconfiguration.

Wi-Fi/802.15.4 coexistence test results show substantial 802.15.4 performance improvements when PTA is utilized:

1. Improved device join success:
  1. However, device join utilizes broadcast messages, which are not retried.
  2. *If possible, device join success can be further improved by temporarily reducing Wi-Fi traffic during devices joining 802.15.4 network.*
2. Substantially reduced MAC retries:
  1. Reduces message latency.
  2. Improves end-node battery life.
  3. Frequency separation remains important, as best managed coexistence performance is for "far-away" channels.
3. Substantially reduced message failure:
  1. 802.15.4 network remains operational, even during high Wi-Fi duty cycles.





Smart.  
Connected.  
Energy-Friendly.



**Products**

[www.silabs.com/products](http://www.silabs.com/products)



**Quality**

[www.silabs.com/quality](http://www.silabs.com/quality)



**Support and Community**

[community.silabs.com](http://community.silabs.com)

#### 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 and limitation to product information, specifications, and descriptions herein, and does not give warranties as to the accuracy or completeness of the included information. Silicon Labs shall have no liability for the consequences of use of the information supplied herein. This document does not imply or express copyright licenses granted hereunder to design or fabricate any integrated circuits. The products are not designed or authorized to be used within any Life Support System 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.

#### Trademark Information

Silicon Laboratories Inc.®, Silicon Laboratories®, Silicon Labs®, SiLabs® and the Silicon Labs logo®, Bluegiga®, Bluegiga Logo®, Clockbuilder®, CMEMS®, DSPLL®, EFM®, EFM32®, EFR, Ember®, Energy Micro, Energy Micro logo and combinations thereof, "the world's most energy friendly microcontrollers", Ember®, EZLink®, EZRadio®, EZRadioPRO®, Gecko®, ISOmodem®, Precision32®, ProSLIC®, Simplicity Studio®, SiPHY®, Telegesis, the Telegesis Logo®, USBXpress® 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. 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

<http://www.silabs.com>