AN1445: SiWT917 RCP Wi-Fi Concurrent Mode

This document details the operation of the SiWT917 RCP Wi-Fi module in concurrent mode. It includes step-by-step instructions for setting up and evaluating the module. Concurrent mode enables the device to function simultaneously as an Access Point (AP) and a Station (STA), allowing users to create separate virtual interfaces for each mode.

KEY POINTS

- Setup Requirements
- Detailed steps for concurrent mode

Table of Contents

1. Introduction

The SiW917 from Silicon Labs supports WiFi Concurrent mode, enabling the device to operate in both Station (STA) and Access Point (AP) modes simultaneously. This dual functionality enhances the versatility and flexibility of WiFi-enabled devices, making them suitable for a wide range of applications.

Devices can connect to the internet to send data to a cloud server (STA mode) while also allowing local devices to connect directly for configuration or control (AP mode).

For more information, refer to the SiWT917 RCP Developers Guide and Getting Started Guide implemented for SiWT917 RCP family of modules, which uses netlink sockets.

2. Prerequisites

Following are the details for the prerequisties required for both hardware and software.

2.1 Hardware

Following are the details for hardware requirements.

Table 2.1. Hardware Requirements

S.N.	Hardware Components	Quantity	Description
1.	SiWT917 RCP Wi-Fi 6 Single Band + BLE 5.4 Wireless Radio. **Radio board:** BRD4346A. **Adapter board:** BRD8045B.	1	1. **SiWx917_RB4346A -** SiWx917 Wi-Fi 6 and Bluetooth LE IC Co-Processor Radio board. 2. **BRD8045B-** Adapter board to mount on Raspberry Pi Expansion Kit (RPI Connector).
2.	PC/Laptop/Embedded Platform with Linux OS	1	Raspberry Pi 4 with SiWT917 RPi image.
3.	Standard WLAN Access Point	1	For Example, TP-Link AX1500 Wi-Fi 6 Router.
4.	Third party Station	1	**Note:** Use third party stations like phones/ tablets/pc/laptops having capabilities to connect to Wi-Fi 6 enabled access points. **Example:** For this test case, OnePlus Nord mobile has been used.
5.	Monitor, mouse, and keyboard	1	To access the console or get the UI access of Raspberry Pi 4.
6.	Ethernet/HDMI cables	1	To connect Raspberry Pi 4 with the monitor.

Note: For more information, follow the: Getting Started Guide.

2.2 Software

Following are the details for software requirements.

Table 2.2. Software Requirements

S.N.	Software Components	Description
1.	SiWT917 RCP Driver	si91x-rcp-driver
2.	Kernel Version from 3.18 to 6.1	For example, In this test case, the system's kernel version is 6.1
3.	wpa supplicant	For example, wpa_supplicant 2.10.
4.	hostapd	**Note:** • Hostapd application version used is v.2.10. • If hostapd is not present in system, give the following command to install hostapd application. **Command:** apt install hostapd .

3. Functional Description SiWT917_EB4346B with Raspberry Pi 4 Platform

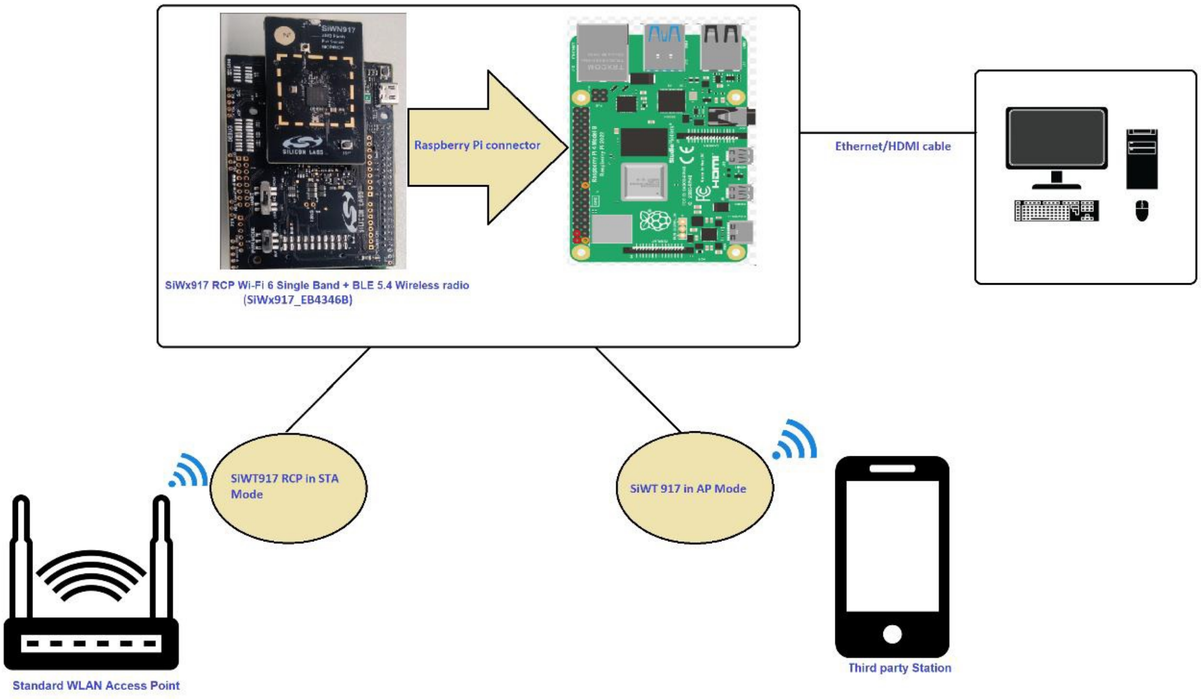


Figure 3.1. Setup Diagram

In the figure above, the user needs to connect the SiWT917 RCP module to a Raspberry Pi 4 running Raspberry Pi 4 OS through the Raspberry Pi connector (40 PIN header) . The Raspberry Pi should have a kernel version installed between 3.18 to 6.1. To evaluate concurrent mode, an external third-party AP and STA are needed.

3.1 Use Cases

- We can use this feature in devices that need wireless third-party access point for internet access and need an admin interface to control and configure the services provided by it.
- Create a bridge/hub with internet access to provide internet access to multiple IoT devices like a Wi-Fi extender.

4. Usage Guidelines

4.1 Steps to bring up in Concurrent Mode

1. Download the SiWT917 Driver.
2. Unzip the driver using the following command.

```
# unzip SiWT917.x.x.x.x.zip
```

3. Next the user needs to enter the root-user mode by giving the following command and providing the correct username and password.

```
# sudo su
```

   The section below provides the steps to configure Wi-Fi Concurrent mode using a startup script or by manual commands. The user can choose either method.

4.1.1 Using Startup Scripts

Use the script at the path “<system_path>/SiWT917.x.x.x.x/release/” to run Wi-Fi concurrent mode.

```
# ./start_SiWT917.sh AP_STA
```

For more details about the startup script file, refer to the Startup Script section of SiWT917 RCP Developer’s Guide.

4.1.2 Using Manual Steps

1. To enable the concurrent mode, the user need to compile the source by enabling the **CONFIG_STA_PLUS_AP** in Makefile at `<system_path>/SiWT917.x.x.x.x/`.

```
#Uncomment below line for using Concurrent mode CONFIG_STA_PLUS_AP = y
```

   Note: `<system_path>` is the location where the user has downloaded/placed the SiWT917 driver in the system.

2. After enabling **CONFIG_STA_PLUS_AP** flag in Makefile, save the file and compile the driver follow.

```
#make clean; make
```

   Note: For compiling from kernel source or for other embedded platforms like iMX6 , the user can refer to the section Compilation Steps.

3. Before installing the driver, install the dependencies using the following commands

```
# modprobe mac80211
# modprobe bluetooth
```

4. Before installation, the user needs to stop the existing network manager and unblock WLAN from rfkill. The commands below are used to stop the network-manager on different Linux distribution.
   - For Ubuntu, use the following command:

```
# service network-manager stop
```

   - For Fedora/Raspberry Pi, use the following command:

```
# service NetworkManager stop
```

   - To stop rfkill blocking WLAN, use the following command :

```
#rfkill unblock wlan (or)
# rfkill unblock all
```

5. Go to the driver package and copy all the files present in the `<system_path>/SiWT917.x.x.x.x/` **Firmware** folder to `/lib/firmware` by following the commands below.

```
# cd <system_path>/SiWT917.x.x.x.x/
# cp Firmware/* /lib/firmware
```

6. After compiling the driver go to `<system_path>/ SiWT917.x.x.x.x/release` folder and give the following commands.

   Enter the following command :

```
# insmod rsi_91x.ko dev_oper_mode = 3 rsi_zone_enabled = 0x601
# insmod sdio.ko sdio_clock = 50
```

7. Check for the interface created using the following command:

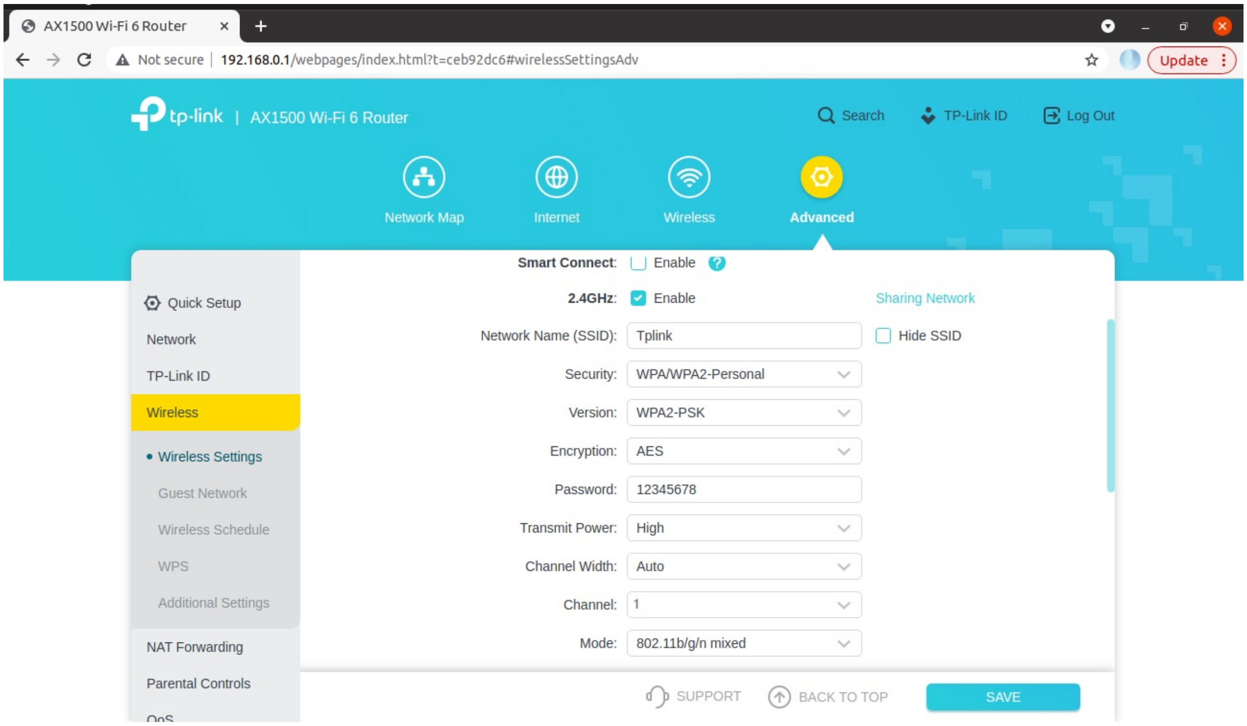
```
# ifconfig -a
```

   For example, if the driver is loaded successfully and the wireless interface is created ,then the user will see the following output :

```
wlan0: flags=4098<BROADCAST,MULTICAST> mtu 1500
        ether 94:b2:16:98:ac:dc txqueuelen 1000 (Ethernet) RX packets 0 bytes 0
        (0.0 B)
        RX errors 0 dropped 0 overruns 0 frame 0
        TX packets 0 bytes 0 (0.0 B)
        TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

   Note: In this test case, the wireless interface created after loading of the driver is “wlan0". The interface name may vary across the systems.

8. Bring up the third-party access point in the desired channel and security. For this test case setup, the **TP-Link AX1500 Wi-Fi 6 Router** is configured with the following credentials as shown in the below figure.



Smart Connect: ☐ Enable

2.4GHz: ☑ Enable — Sharing Network

Network Name (SSID): Tplink ☐ Hide SSID

Security: WPA/WPA2-Personal

Version: WPA2-PSK

Encryption: AES

Password: 12345678

Transmit Power: High

Channel Width: Auto

Channel: 1

Mode: 802.11b/g/n mixed

Figure 4.1. TP-Link AX1500 Wi-Fi 6 Router

9. Edit the network block present in the `<system_path>/SiWT917.x.x.x.x/release/ sta_settings.conf` file present in the `<system_path>/ SiWT917.x.x.x.x/release` folder with the credentials of the third-party WLAN access point. For this test case, the network block is updated in the following manner:

```
ctrl_interface = /var/run/wpa_supplicant
         update_config = 1
         #Enable this network block for CCMP/TKIP mode
         network = {
         ssid = "Tplink"
        pairwise = CCMP TKIP
        group = CCMP TKIP
        key_mgmt = WPA-PSK psk = "12345678"
        # bgscan = "simple:15:-45:20"
        proto = WPA2 WPA
        }
```

For more details regarding how to update the network block for other security modes in `<system_path>/ SiWT917.x.x.x.x/ release/sta_settings.conf file`, the user needs to follow the section **Configure Station Using WPA Supplicant** of the SiWT917 RCP Developer's Guide.

10. Run wpa_supplicant to connect SiWT917-STA to the TAP.

```
#wpa_supplicant -i <interface_name> -D nl80211 -c
        <system_path>/SiWT917.x.x.x.x/release/sta_settings.conf -ddddt > log &
        Example : wpa_supplicant -i wlan0 -D nl80211 -c /home/
        SiWT917.x.x.x.x/release/sta_settings.conf -ddddt > supp.log &
```

11. To check whether the connection is successful or not use below command:

```
# iwconfig
```

If the connection is successful, then the connected access point SSID along with the MAC address is displayed as shown below.

```
wlan0 IEEE 802.11 ESSID:"Tplink"
Mode:Managed Frequency:2.412 GHz Access Point: B0:A7:B9:C4:52:CA
Bit Rate = 39 Mb/s Tx-Power = 16 dBm
Retry short limit:7 RTS thr = 2353 B Fragment thr=2352 B
Encryption key:off
Power Management:off
Link Quality = 80/80 Signal level = -28 dBm
Rx invalid nwid:0 Rx invalid crypt:0 Rx invalid frag:0
Tx excessive retries:0 Invalid misc:18 Missed beacon:0
```

If it is not connected to an access point, a message **Not Associated** is displayed as shown below.

```
wlan0 IEEE 802.11 ESSID:off/any
Mode: Managed
Access Point: Not-Associated Tx-Power = 0 dBm Retry short limit:7 RTS thr:off Fragment thr:off
Encryption key:off
Power Management:off
```

12. After successful connection, check the IP address using the below commands

```
# dhclient wlan0 -r
# dhclient wlan0 -v
```

13. To check if the SiWT917-STA has assigned an IP address from the third-party wlan access point, the user can give the following command:

    ```
    #ping <IP_adrress of TAP>
             Example : ping 192.168.0.1
    ```

    For example, if SiWT917-STA has successfully received an IP, we will see the following output.

    ```
    PING 192.168.0.1 (192.168.0.1) from 192.168.0.228 wlan0:
    56(84) bytes of data.
    64 bytes from 192.168.0.1: icmp_seq = 1 ttl = 64 time = 26.8 ms
    64 bytes from 192.168.0.1: icmp_seq = 2 ttl = 64 time = 10.8 ms
    64 bytes from 192.168.0.1: icmp_seq = 3 ttl = 64 time = 4.00 ms
    64 bytes from 192.168.0.1: icmp_seq = 4 ttl = 64 time = 6.25 ms
    64 bytes from 192.168.0.1: icmp_seq = 5 ttl = 64 time = 1.77 ms
    64 bytes from 192.168.0.1: icmp_seq = 6 ttl = 64 time = 5.05 ms
    64 bytes from 192.168.0.1: icmp_seq = 7 ttl = 64 time = 2.18 ms
    64 bytes from 192.168.0.1: icmp_seq = 8 ttl = 64 time = 5.63 ms
    64 bytes from 192.168.0.1: icmp_seq = 9 ttl = 64 time = 2.72 ms
    64 bytes from 192.168.0.1: icmp_seq = 10 ttl = 64 time = 3.01 ms
    64 bytes from 192.168.0.1: icmp_seq = 11 ttl = 64 time = 2.32 ms
    64 bytes from 192.168.0.1: icmp_seq = 12 ttl = 64 time = 3.14 ms
    --- 192.168.0.1 ping statistics ---
    12 packets transmitted, 12 received, 0 % packet loss, time 11019 ms
    rtt min/avg/max/mdev = 1.766/6.133/26.773/6.665 ms
    ```

    For example, if SiWT917-STA has not assigned with an IP address, we will see the following output for the ping command.

    ```
    # ping: connect: Network is unreachable
    ```

14. Create the AP vap, using the following command:

    ```
    # iw dev wlan0 interface add wlan1 type __ap
    ```

15. Check the interface name created by using the following command:

    ```
    # ifconfig -a
    ```

    If the interface is successfully created , we will get the following output :

    ```
    wlan1: flags = 4098<BROADCAST,MULTICAST> mtu 1500
    ether 94:b2:16:98:ac:dd txqueuelen 1000 (Ethernet)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0 TX packets 0 bytes 0 (0.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    ```

    Note: In this test case the interface name for SiWT917-AP is created as "wlan1". The naming convention is system specific. The user can get the same name or a different name depending upon the target host.

16. Configure the fields present in ap_open.conf or ap_wpa.conf file and bring up RSI-AP as follows that is, change the **interface** field value present in `<system_path>/SiWT917.x.x.x.x/release/  /ap_open.conf` file or `<system_path>/SiWT917.x.x.x.x/release/ap_wpa.conf` file with the new interface name created for AP vap.

    For example, we brought the RSI-AP in open security mode with the following credentials.

```
interface = wlan1
driver = nl80211
ctrl_interface = /var/run/hostapd
ctrl_interface_group = 0

ssid=bionic_test ignore_broadcast_ssid = 0
hw_mode = g channel = 1 beacon_int = 100
dtim_period = 2
max_num_sta = 4
rts_threshold = 2347
fragm_threshold = 2346
auth_algs = 1
# Country Related
#ieee80211d = 1
country_code = IN
wmm_enabled = 1
wmm_ac_bk_cwmin = 4
wmm_ac_bk_cwmax = 10
wmm_ac_bk_aifs = 7
wmm_ac_bk_txop_limit = 0
wmm_ac_bk_acm = 0
wmm_ac_be_aifs = 3
wmm_ac_be_cwmin=4 wmm_ac_be_cwmax = 10
wmm_ac_be_txop_limit = 0 wmm_ac_be_acm = 0
wmm_ac_vi_aifs = 2
wmm_ac_vi_cwmin = 3
wmm_ac_vi_cwmax = 4
wmm_ac_vi_txop_limit = 94
wmm_ac_vi_acm = 0
wmm_ac_vo_aifs = 2
wmm_ac_vo_cwmin   = 2
wmm_ac_vo_cwmax = 3
wmm_ac_vo_txop_limit = 47
wmm_ac_vo_acm = 0
eap_server  = 0
```

    Then bring up the ap_vap with the following command:

```
# hostapd ap_open.conf -dddt >log1 &
```

17. To check whether the AP is up or not, use the following command:

```
# iw dev
```

    For example, if bringing up of the AP mode is successful, we will see the following output.

```
phy#1
Interface wlan1
    ifindex 12
    wdev 0x400000002 addr 94:b2:16:98:ac:dd
    ssid bionic_test
    type AP
    channel 1 (2412 MHz), width: 20 MHz (no HT), center1: 2412 MHz
    txpower 20.00 dBm
Interface wlan0
    ifindex 9 wdev
    0x300000001
    addr 94:b2:16:98:ac:dc
    ssid Tplink type managed
    channel 1 (2412 MHz), width: 20 MHz, center1: 2412 MHz
    txpower 20.00 dBm
```

    In the example above, we can see for **Interface wlan1** ,the type is **AP**. The user can now check the SiWT917-AP is up with the ssid **bionic_test** .

18. Run the dhcp server for AP vap.

```
# sh dhcp_server.sh wlan1
```

19. Connect third-party STA to SiWT917-AP. For example, you can see the below image:

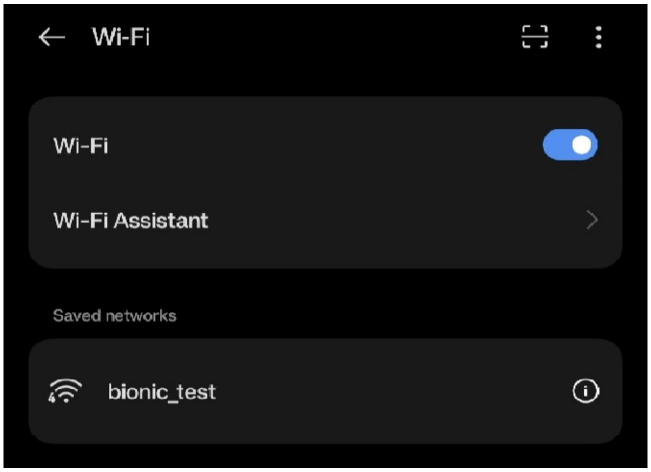


Figure 4.2. Connection to third-party STA

4.2 Limitations

Following are the limitations:

- Always start SiWT917-STA first, let the SiWT917-STA connection happen to TAP, and then start the SiWT917-AP mode.
- In concurrent mode, if SiWT917-STA interface goes down, then SiWT917-AP interface must be put down to restart the SiWT917-STA mode.
- SiWT917-STA cannot use radio for scanning once it is acquired by SiWT917-AP for beacon emission for regular interval.
- Background scan(bg-scan) and power save features are not supported for the station mode vap in concurrent mode.
- SiWT917-AP will always operate in channel in which the SiWT917-STA [corresponding to other VAP] connects. For example, if the station connects in channel 6, then AP mode should be created in channel 6, irrespective of the channel configured; however, SiWT917-AP and SiWT917-STA can operate in different security modes.

5. Summary/ Conclusion

By following the steps outlined in this document, the user can bring up SiWT917 RCP module in concurrent mode with any Linux based host platform.

6. Appendix A: Terminology

Common acronyms and abbreviations used in this document:

- **AP -** Access Point.
- **STA -** Station.
- **TAP -** Third party WLAN Access Point.
- **SiWT917-STA -** Station interface that is created for SiWT917 RCP after loading the driver.
- **SiWT917-AP -** Access Point interface that is created for SiWT917 RCP after loading the driver.

7. Appendix B: Refrences and Related Documentation

1. Refer to SiWT917 RCP Developers Guide and Getting Started Guide.
2. Refer to Concurrent mode.

8. Appendix C: Troubleshooting

- Make sure the third-party AP is up while connecting to it. Also make sure to add the valid credentials depending on security type in the `<system_path>/SiWT917.x.x.x.x/release/sta_settings.conf` file while running the supplicant.
- Make sure that the AP should be brought in same channel in which the STA is connected to the third-party AP.

9. Revision History

Revision 1.3

July 2025

4.1.2 Using Manual Steps: Following are the updates:

- Replace Ubuntu/Raspberry Pi with Ubuntu
- Replace Fedora with Fedora/Raspberry Pi

Revision 1.2

March 2025

- Updated dev_oper_mode = 3 from dev_oper_mode = 1 in point 6 in 4.1.2 Using Manual Steps.

Revision 1.1

January 2025

- Removed BRD4357A radio board reference from Table 2.1 Hardware Requirements on page 4.

Revision 1.0

January 2025

- Initial release.

IoT Portfolio
www.silabs.com/products

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.

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 LABS

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

www.silabs.com