



# AN1436: SiWx917 QMS Crystal Calibration

## Application Note

---

This application note describes the crystal calibration procedure for the SiWx917 QMS package to arrive at the right value for carrier frequency offset and updating the optimized values to the flash of the chip. Customers are required to calibrate frequency offsets on their boards after the chip is assembled with the planned front-end circuit. This document covers the commands used to determine the frequency offset and the flow of measurement and correction.

### KEY POINTS

---

- Prerequisites, Setup Requirements, and Setup Diagrams
- Frequency Offset Measurements
- Frequency Offset Corrections

# Table of Contents

<b>1. Introduction</b>	<b>3</b>
<b>2. Prerequisites and Setup Requirements</b>	<b>4</b>
2.1 Hardware	4
2.1.1 SoC Mode	4
2.1.2 NCP_Mode	4
2.2 Software	5
<b>3. Setup Diagram</b>	<b>6</b>
3.1 SoC Mode	6
3.2 NCP Mode	6
<b>4. Setting-up the Project Environment</b>	<b>7</b>
4.1 Creating the Project	7
4.2 Configuring the Application	7
4.3 Implementing CW Mode	8
4.4 Application Execution Flow	8
<b>5. Frequency Offset Measurement</b>	<b>9</b>
5.1 Measure with Transmitted Continuous Wave (CW)	9
5.2 Measure with Transmitted Burst	10
<b>6. Frequency Offset Correction</b>	<b>11</b>
6.1 Parameters	13
<b>7. Calibration Using Manufacturing Utility</b>	<b>14</b>
7.1 Transmission in Burst Mode	14
7.2 Transmission in Continuous Wave Mode	14
<b>8. Acronyms and Abbreviations</b>	<b>15</b>
<b>9. Revision History</b>	<b>16</b>

## 1. Introduction

The crystal calibration can be done with two approaches. The first approach uses a continuous wave (CW) DC tone unmodulated carrier for transmission and the second approach uses Burst Mode, which transmits modulated data. The first approach only requires a normal spectrum analyzer. Burst mode can only be used if test instrument supports modulation analysis where frequency error can be reported by instrument.

The sequence of steps includes the following:

- Start transmission in packet burst or CW mode that allows control of data rate, transmit power and frame length.
- Measure the carrier frequency offset.
- Update the calibrated parameters to chip.

## 2. Prerequisites and Setup Requirements

### 2.1 Hardware

- Host MCU Silicon Labs WPK
- Windows PC
- Spectrum Analyzer
- RF Cable (or antenna) to connect the Radio/Exp Board and the spectrum Analyzer
- There are two hardware modes:
  - SoC Mode
  - NCP Mode

#### 2.1.1 SoC Mode

Following are the details for SoC mode:

- Standalone
  - BRD4002A Wireless pro kit mainboard [SI-MB4002A]
  - Radio Boards
    - BRD4338A [SiWx917-RB4338A]
    - BRD4342A [SiWx917-RB4342A]
    - BRD4343A [SiWx917-RB4343A]
- Kits
  - SiWx917 Pro Kit [[Si917-PK6031A](#)]
  - SiWx917 Pro Kit [Si917-PK6032A]

#### 2.1.2 NCP\_Mode

Following are the details for NCC mode:

- Standalone
  - BRD4002A Wireless pro kit mainboard [SI-MB4002A]
  - EFR32xG24 Wireless 2.4 GHz +10 dBm Radio Board [[xG24-RB4186C](#)]
  - NCP Expansion Kit with NCP Radio boards
    - (BRD4346A + BRD8045A) [SiWx917-EB4346A]
    - (BRD4357A + BRD8045A) [SiWx917-EB4357A]
- Kits  
EFR32xG24 Pro Kit +10 dBm [xG24-PK6009A](#)
- STM32F411RE MCU
  - [STM32F411RE](#)
  - NCP Expansion Kit with NCP Radio boards
    - (BRD4346A + BRD8045C)
    - (BRD4357A + BRD8045C)
- Interface and Host MCU Supported
  - SPI - EFR32 and STM32
  - UART - EFR32

## 2.2 Software

- Simplicity Studio IDE
  - Download the [Simplicity Studio IDE](#).
  - Follow the [Simplicity Studio User Guide](#) to install the Simplicity Studio IDE.
  - Install a SiSDK suite with the WiSeConnect SDK included.
  - Update the SiWx91x Connectivity Firmware.

**Note:** The WiSeConnect and Connectivity Firmware installation is described here in detail: [Getting Started with the WiSeConnect SDK](#).

### 3. Setup Diagram

#### 3.1 SoC Mode

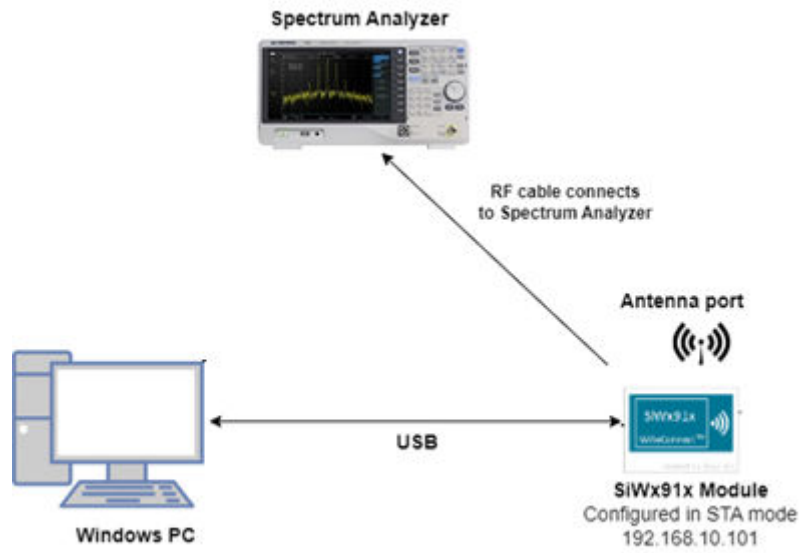


Figure 3.1. Calibration Test Setup for SoC Mode

#### 3.2 NCP Mode

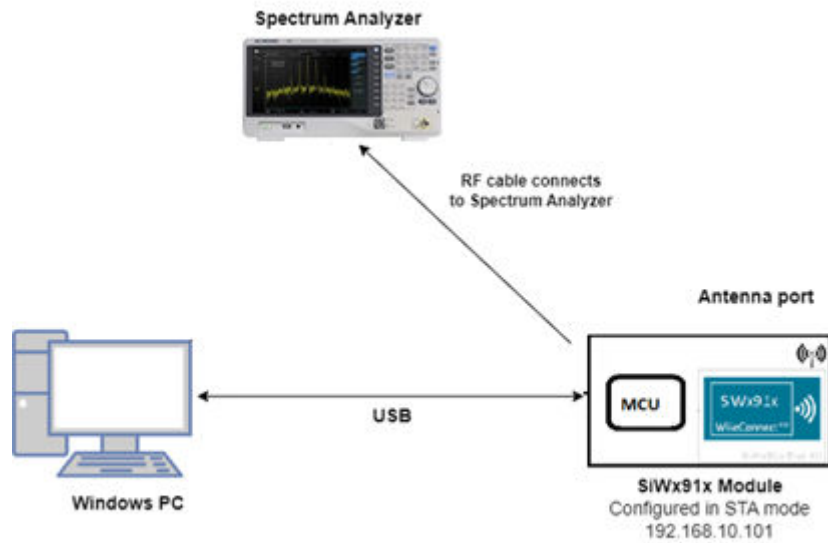


Figure 3.2. Calibration Test Setup for NCP Mode

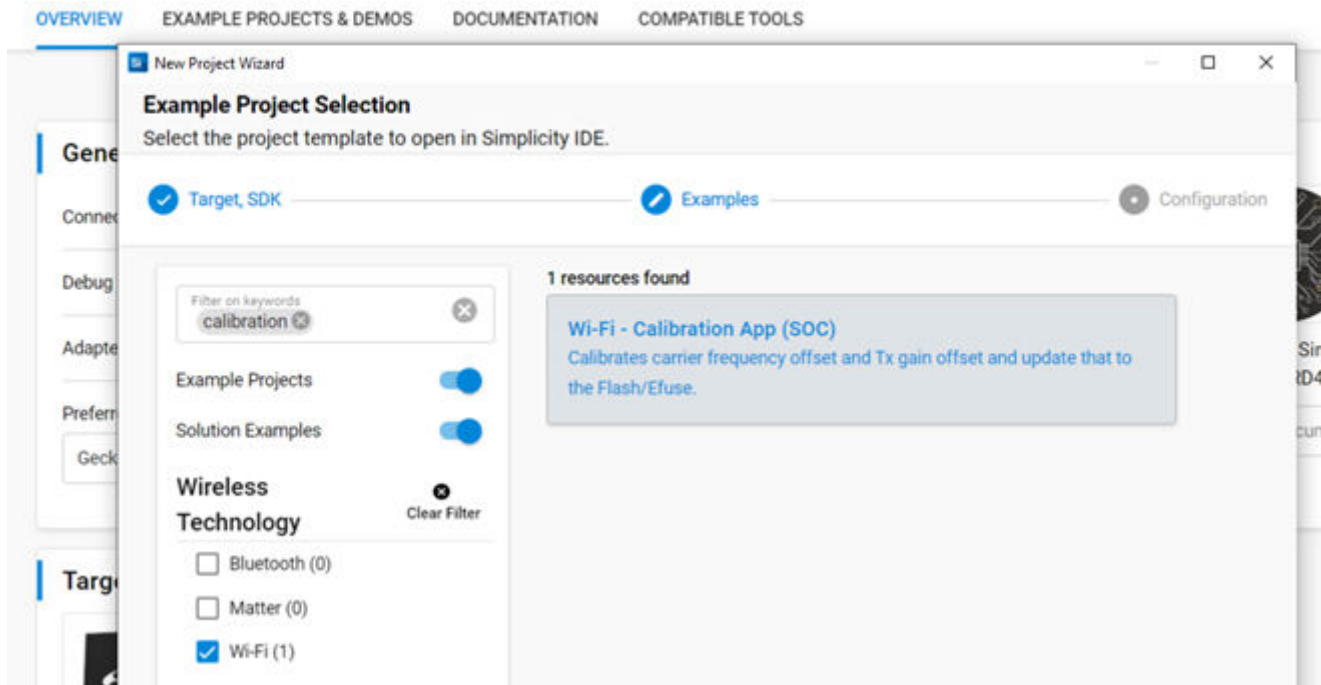
## 4. Setting-up the Project Environment

### 4.1 Creating the Project

The calibration can be done easily with the Wi-Fi calibration application provided by the WiSeConnect SDK. Project configuration and running the calibration example is described here in more details: [Project Environment](#).

To get started, create a Wi-Fi Calibration (SoC or NCP) application. Open the new project wizard in Simplicity studio and choose the correct application.

### SiWG917 Single Band Wi-Fi and BLE 8MB Flash RB, Wireless Pro Kit Mainboard (ID: 0004



### 4.2 Configuring the Application

Once the application project is created, the application can be configured to suit the user requirements and the development environment. Read through the following sections and desired modifications.

Configure the following parameters in **app.c** to test Xtal Calibration app as per requirements:

```
#define TX_TEST_POWER      18 // Sets TX power in dBm. The valid values are from (2 to 18)dBm and 127.
#define TX_TEST_RATE      0 // Sets transmit data rate.
#define TX_TEST_LENGTH    30 // Configures length of the TX packet. Valid values are in the range of 24 to
1500 bytes in the burst mode.
#define TEST_CHANNEL      1 // Configures the channel number in 2.4 GHz Band
#define TX_TEST_MODE      BURST_MODE // Configures Burst mode, Continuous mode or CW mode
```

#### Note:

- Tx burst mode can only be used if test instrument supports Modulation analysis measurement, where the Freq error can be reported by the instrument.
- Tx CW mode is not implemented in this example. To access CW mode, modify the app.c file based as mentioned in the following section.

### 4.3 Implementing CW Mode

1. Define and set CW\_MODE as a Tx test mode in the constants section.

```
#define BURST_MODE          0
#define CONTINUOUS_MODE    1
#define CW_MODE            2    // CW/DC unmodulated Carrier mode is defined here

#define TX_TEST_POWER      18
#define TX_TEST_RATE       0
#define TX_TEST_LENGTH     30
#define TX_TEST_MODE       CW_MODE    // Test mode is changed here
```

2. Start a burst generation, stop it and start the CW mode generation in the application\_start() function. Add the next lines just before sl\_si91x\_transmit\_test\_start() is called.

```
status = sl_si91x_transmit_test_start(TX_TEST_POWER, TX_TEST_RATE, TX_TEST_LENGTH, 0, TEST_CHANNEL);
if (status != SL_STATUS_OK) {
    printf("Transmit test start failed: 0x%lx\r\n", status);
} else {
    printf("Transmit test started\r\n");
}

osDelay(1000);

status = sl_si91x_transmit_test_stop();

osDelay(1000);
```

3. Build the code.
4. After confirming the input parameters, build and flash the application.

### 4.4 Application Execution Flow

1. Connect WPK Mainboard via USB to the computer and flash the compiled project to the board.
2. Launch a console on the device to access the command line interface. After the program gets executed (device is reset or powered up), the SiWx91x device will start the transmit test with the given configuration. Application prints (over JLink port) would be as follows:

```
9/13/2023 13:23:01.513 [RX] - <NULL>
9/13/2023 13:23:05.432 [RX] - Wi-Fi initialization successful<CR><LF>
Transmit test started<CR><LF>
Calibration commands usage:<CR><LF>
.....<CR><LF>
sl_freq_offset=<freq_offset_in_KHz><CR><LF>
sl_calib_write=<target>,<flags>,<gain_offset_low>,<gain_offset_mid>,<gain_offset_high><CR><LF>
OR<CR><LF>
sl_calib_write=<target>,<flags>,<gain_offset_low>,<gain_offset_mid>,<gain_offset_high>,<cxo_ctune><CR><LF>
sl_evm_offset=<index>,<evm_offset><CR><LF>
sl_evm_write=<target>,<flags>,<evm_offset_11B>,<evm_offset_11G_30M_54M_11N_MCS3_MCS7>,<evm_offset_11G_6M_24M_11N_MCS0_MCS2>,<evm_offset_11N_MCS0>,<evm_offset_11N_MCS7><CR><LF>
.....<CR><LF>
Enter the calibration command:<CR><LF>
```

3. The calibration commands (discussed in [Frequency Offset Correction](#)) can be entered in the serial terminal as input.

```
9/13/2023 13:57:00.421 [RX] - Wi-Fi initialization successful<CR><LF>
Transmit test started<CR><LF>
Calibration commands usage:<CR><LF>
.....<CR><LF>
sl_freq_offset=<freq_offset_in_KHz><CR><LF>
sl_calib_write=<target>,<flags>,<gain_offset_low>,<gain_offset_mid>,<gain_offset_high><CR><LF>
OR<CR><LF>
sl_calib_write=<target>,<flags>,<gain_offset_low>,<gain_offset_mid>,<gain_offset_high>,<cxo_ctune><CR><LF>
sl_evm_offset=<index>,<evm_offset><CR><LF>
sl_evm_write=<target>,<flags>,<evm_offset_11B>,<evm_offset_11G_30M_54M_11N_MCS3_MCS7>,<evm_offset_11G_6M_24M_11N_MCS0_MCS2>,<evm_offset_11N_MCS0>,<evm_offset_11N_MCS7><CR><LF>
.....<CR><LF>
Enter the calibration command:<CR><LF>

9/13/2023 14:00:53.154 [TX] - sl_freq_offset=2<CR><LF>
9/13/2023 14:00:53.166 [RX] - Command read complete<CR><LF>
Frequency offset correction successful<CR><LF>
Calibration commands usage:<CR><LF>
.....<CR><LF>
sl_freq_offset=<freq_offset_in_KHz><CR><LF>
sl_calib_write=<target>,<flags>,<gain_offset_low>,<gain_offset_mid>,<gain_offset_high><CR><LF>
OR<CR><LF>
sl_calib_write=<target>,<flags>,<gain_offset_low>,<gain_offset_mid>,<gain_offset_high>,<cxo_ctune><CR><LF>
sl_evm_offset=<index>,<evm_offset><CR><LF>
sl_evm_write=<target>,<flags>,<evm_offset_11B>,<evm_offset_11G_30M_54M_11N_MCS3_MCS7>,<evm_offset_11G_6M_24M_11N_MCS0_MCS2>,<evm_offset_11N_MCS0>,<evm_offset_11N_MCS7><CR><LF>
.....<CR><LF>
Enter the calibration command:<CR><LF>

9/13/2023 14:00:55.020 [TX] - sl_calib_write=1,2,0,0,0<CR><LF>
9/13/2023 14:00:55.030 [RX] - Command read complete<CR><LF>
```



## 5. Frequency Offset Measurement

### 5.1 Measure with Transmitted Continuous Wave (CW)

1. Start up the example application in CW mode and observe the frequency error difference between the calibrated spectrum analyzer and the SoC as indicated below (a marker could be useful to make accurate readings.)

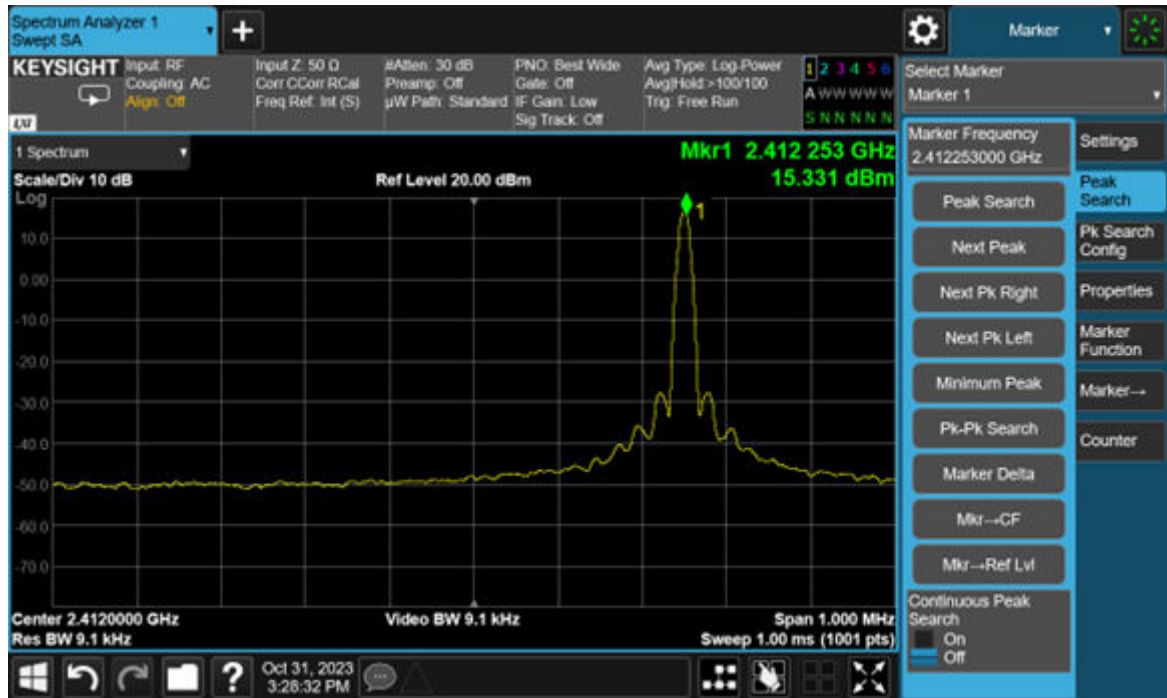


Figure 5.1. Frequency Offset Measured in CW Mode with a Traditional Spectrum Analyzer

2. Note the frequency offset, which will be fed back to the device in the following section.

## 5.2 Measure with Transmitted Burst

1. Start up the example application in burst mode and set up the spectrum analyzer in a way that the frequency offset can be measured. Here's an example of such measurement:

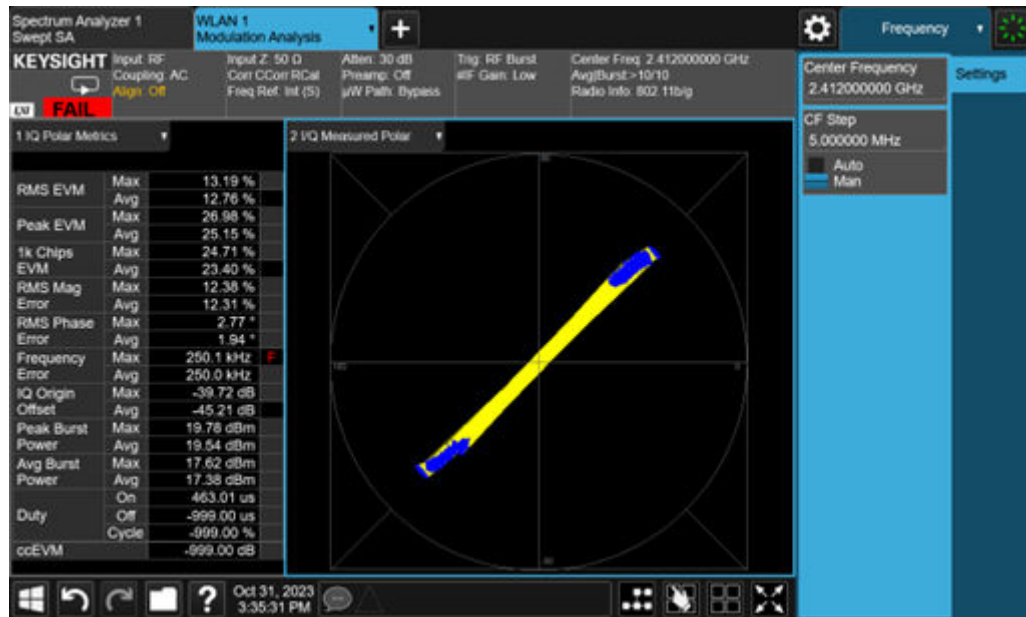


Figure 5.2. Frequency Offset Measured in Burst Mode with a Traditional Spectrum Analyzer

**Note:** In the above measurement example, for both CW and Burst Mode, a Frequency error of ~250 KHz is reported by Spectrum Analyzer.

## 6. Frequency Offset Correction

Frequency offset correction is done by using the `sl_freq_offset` command. This command is used during the RF calibration process and requires PER mode transmissions to be initiated prior. This command sets `freq_offset` (deviation) as observed on the signal analyzer against the expected channel frequency.

Frequency offset correction command syntax:

```
sl_freq_offset = freq_offset_in_kHz  
<CR><LF>
```

Examples:

If an error of 10 KHz is observed on the instrument, the command should be used as follows:

```
sl_freq_offset=10<CR><LF>
```

If an error of -10 KHz is observed on the instrument the command should be used as follows:

```
sl_freq_offset=-10<CR><LF>
```

With the above example measurements in CW and Burst Mode, a Frequency Error of ~250 KHz is observed:

```
sl_freq_offset=-250<CR><LF>
```

**Note:** The user can use the above command multiple times to correct the frequency error till it gets tuned to desired frequency offset.

The user can provide the input to correct the frequency error by sending the commands to the console. This should lead towards a correction in the frequency offset as observed earlier and repeated till the error is within the tolerance limits ( $\pm 2$  KHz tolerance limits).

After the frequency error is corrected, the impact may be observed by a spectrum analyzer measurement. If the results still doesn't meet the requirement, a new round of calibration can be processed.

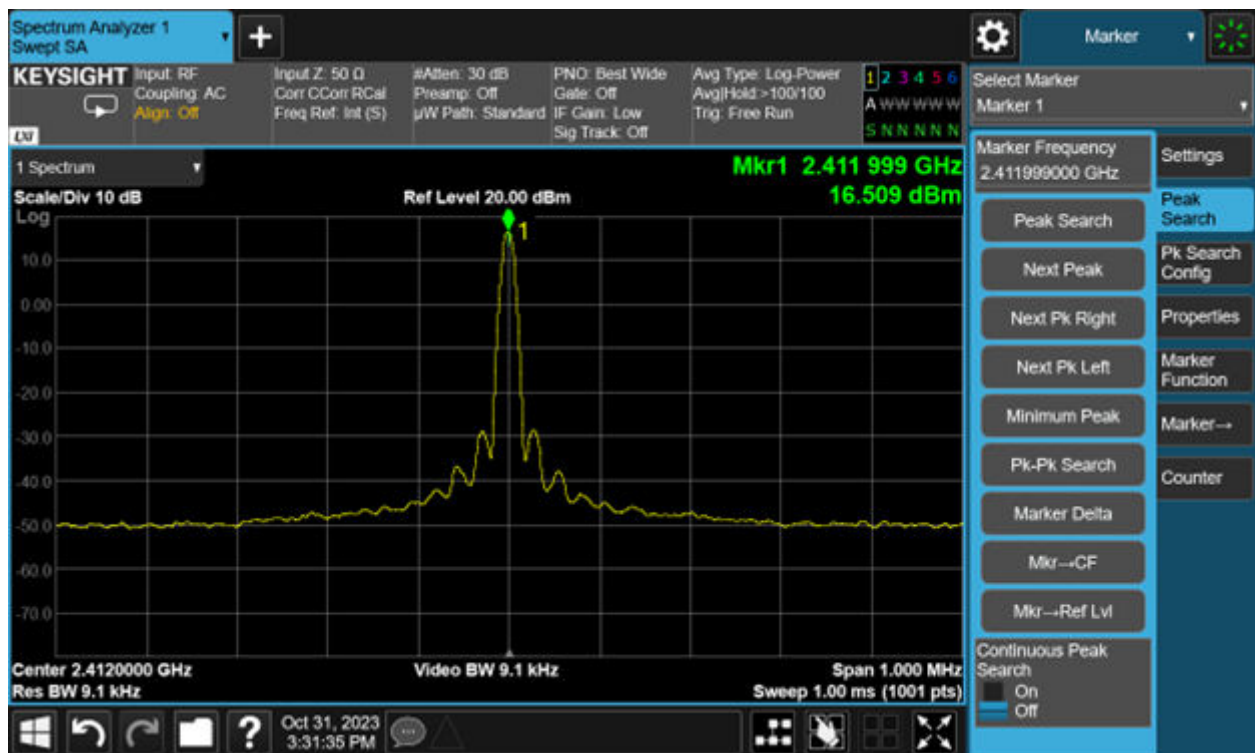


Figure 6.1. Corrected Frequency Offset Measured in CW Mode

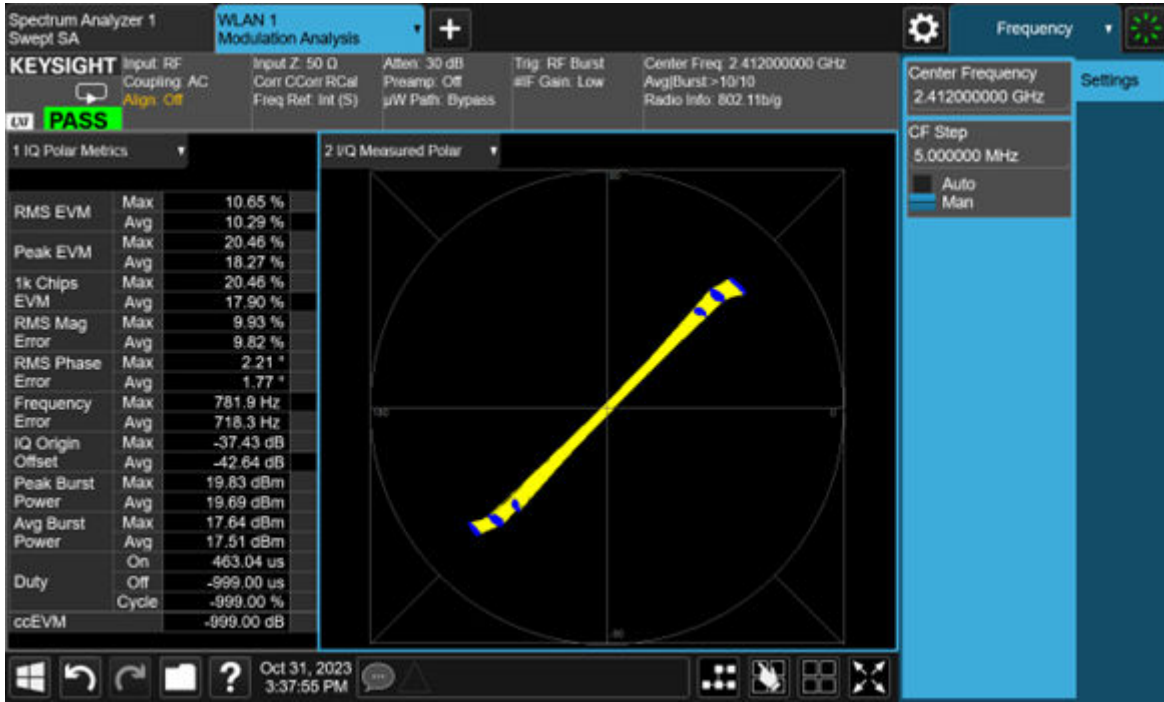


Figure 6.2. Corrected Frequency Offset Measured in Burst Mode

The corrected value of frequency offset is stored within a hardware register and for the correction to reflect upon reset, it is required that we copy the corrected value to flash. The value can be written to flash using the `s1_calib_write` command. The command should be used as follows: `s1_calib_write = 1,2,0,0,0`. Further details regarding the syntax of the `s1_calib_write` command are provided in the subsequent sections.

Using the `s1_calib_write` command, the calibrated XO Ctune and calculated gain offset can be updated to target memory (Flash/Efuse). Adjusting the gain offsets increases/reduces the average Tx power channel power in a particular channel. For further information about gain offset calibration, read through the gain offset calibration guide.

`s1_calib_write` command syntax:

`s1_calib_write=<target>,<flags>,<gain_offset_low>,<gain_offset_mid>,<gain_offset_high>`

OR

`s1_calib_write=<target>,<flags>,<gain_offset_low>,<gain_offset_mid>,<gain_offset_high>,<xo_ctune>`

## 6.1 Parameters

Parameter	Description		
target	<b>Value</b>	<b>Macro</b>	<b>Description</b>
	0	BURN_INTO_EFUSE	Burns calibration data to EFuse
	1	BURN_INTO_FLASH	Burns calibration data to Flash
flags	<b>BIT</b>	<b>Macro</b>	<b>Description</b>
	0		Reserved for future use
	1	BURN_FREQ_OFFSET	1 - Update XO Ctune to calibration data 0 - Skip XO Ctune update
	2	SW_XO_CTUNE_VALID	1- Use XO Ctune provided as argument to update calibration data 0 -Use XO Ctune value as read from hardware register
	3	N/A	
xo_ctune	This field allows the user to directly update xo_ctune value to calibration data bypassing the freq offset loop, valid only when BURN_FREQ_OFFSET & SW_XO_CTUNE_VALID of flags is set. The range of xo_ctune is [0, 255], and the typical value is 80.		

**Precondition:** sl\_freq\_offset command needs to be called before this command, when xo ctune value from hardware register is to be used.

## 7. Calibration Using Manufacturing Utility

The following two flows are possible with the manufacturing utility (Simplicity Commander CLI) to calibrate the SiWG917 device.

### 7.1 Transmission in Burst Mode

#### Steps for CTUNE adjustments

1. Setup the radio and start transmission.

```
commander manufacturing radio --power 16 --phy 6MBPS --channel 1 --start -d SiWG917M111MGTBA
```

2. Measure the frequency on the instrument.
3. Adjust the CTUNE values.

Offset = measured frequency (in kHz) – 2412000. This is a frequency offset correction value ranging from -255 to 255.

```
commander manufacturing xocal --offset --skipload -d SiWG917M111MGTBA
```

4. Check the instrument and verify that the channel frequency is within expectations. If not, repeat step 3.
5. Store the CTUNE values. The radio transmission will stop.

```
commander manufacturing xocal --store --skipload -d SiWG917M111MGTBA
```

### 7.2 Transmission in Continuous Wave Mode

#### Steps for CTUNE adjustments

1. Setup the radio and start transmission.

```
commander manufacturing radio --power 16 --phy CW --channel 1 --start -d SiWG917M111MGTBA
```

2. Measure the frequency on the instrument using peak search.
3. Adjust the CTUNE values.

Offset = measured frequency (in kHz) -- 2412000

```
commander manufacturing xocal --offset --skipload -d SiWG917M111MGTBA
```

4. Check the instrument and verify that the channel frequency is within expectations. If not, repeat step 3.
5. Store the CTUNE values. The radio transmission will stop.

```
commander manufacturing xocal --store --skipload -d SiWG917M111MGTBA
```

#### Note:

1. User can use the above command multiple times to correct the frequency error till it gets tuned to desired frequency offset.
2. User can provide input to correct frequency error by sending the commands on console. This should lead towards a correction in the frequency offset as observed earlier and repeat till the error is within the tolerance limits (+/- 2 KHz tolerance limits).
3. After the frequency error is corrected, the impact may be observed by spectrum analyzer measurement. If the results still don't meet the requirement a new round of calibration can be processed.
4. It is desired to perform calibration for Channel 1 alone. Other channels (especially Channel 6) are sensitive to 40 MHz harmonics, so there is some biasing done internally to reduce its impact. This could lead to variation in frequency offsets, and hence these channels are not suggested.
5. A frequency offset of ~2 KHz (close to 1 ppm) is expected after calibration at room temperature. However, across channels the error can be upto 4 ppm(8-10 KHz) at room temperature.
6. Across the operating temperature range of -40 to 85 °C, the frequency drift can be as much as +/-15 ppm that is, 36 KHz.

## 8. Acronyms and Abbreviations

Acronym	Description
Tx	Transmit
Rx	Receive
RF	Radio Frequency
WLAN	Wireless Local Area Network
XO	Crystal Oscillator
Ctune	Capture

## 9. Revision History

### Revision 1.1

October 2024

- Updated: [2. Prerequisites and Setup Requirements](#).
- Updated: [4.4 Application Execution Flow](#).
- Added: [7. Calibration Using Manufacturing Utility](#).

### Revision 1.0

December 2023

- Initial version.



# Smart. Connected. Energy-Friendly.



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



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



**Support & Community**  
[www.silabs.com/community](http://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.<sup>®</sup>, Silicon Laboratories<sup>®</sup>, Silicon Labs<sup>®</sup>, SiLabs<sup>®</sup> and the Silicon Labs logo<sup>®</sup>, Bluegiga<sup>®</sup>, Bluegiga Logo<sup>®</sup>, EFM<sup>®</sup>, EFM32<sup>®</sup>, EFR, Ember<sup>®</sup>, Energy Micro, Energy Micro logo and combinations thereof, “the world’s most energy friendly microcontrollers”, Redpine Signals<sup>®</sup>, WiSeConnect, n-Link, EZLink<sup>®</sup>, EZRadio<sup>®</sup>, EZRadioPRO<sup>®</sup>, Gecko<sup>®</sup>, Gecko OS, Gecko OS Studio, Precision32<sup>®</sup>, Simplicity Studio<sup>®</sup>, Telegesis, the Telegesis Logo<sup>®</sup>, USBXpress<sup>®</sup>, Zentri, the Zentri logo and Zentri DMS, Z-Wave<sup>®</sup>, 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](http://www.silabs.com)