AN1433: SiWx917 NCP Low Power Application Note

Power is critical for any battery-operated wireless device. Based on the user application, the wireless device can be idle for more time or for a very short time. During this idle period, the wireless devices can be set to sleep to conserve battery power. The SiWx917 wireless device is an optimal ultra-low power wireless (WLAN and Bluetooth) solution for developing applications requiring long battery life.

This application note describes the SiWx917 Power Save Modes, and how to choose Power Save Modes based on application requirements and current consumption analysis during various Operational Modes in NCP mode. It also explains how to configure SiWx917 in different Power Save Modes using power save examples in the WiSeConnect™ SDK v3.x and power optimization techniques that reduce SiWx917's current consumption.

KEY POINTS

- SiWx917's current consumption during various Operational Modes
- Refer to Power Save Examples in the WiSeConnect™ SDK
- Choosing Power Save Modes as per application requirements
- Current measurement methods
- Power Optimization Techniques

1. Power Domains

The following block diagram illustrates various power domains of the SiWx917 chip in NCP mode.

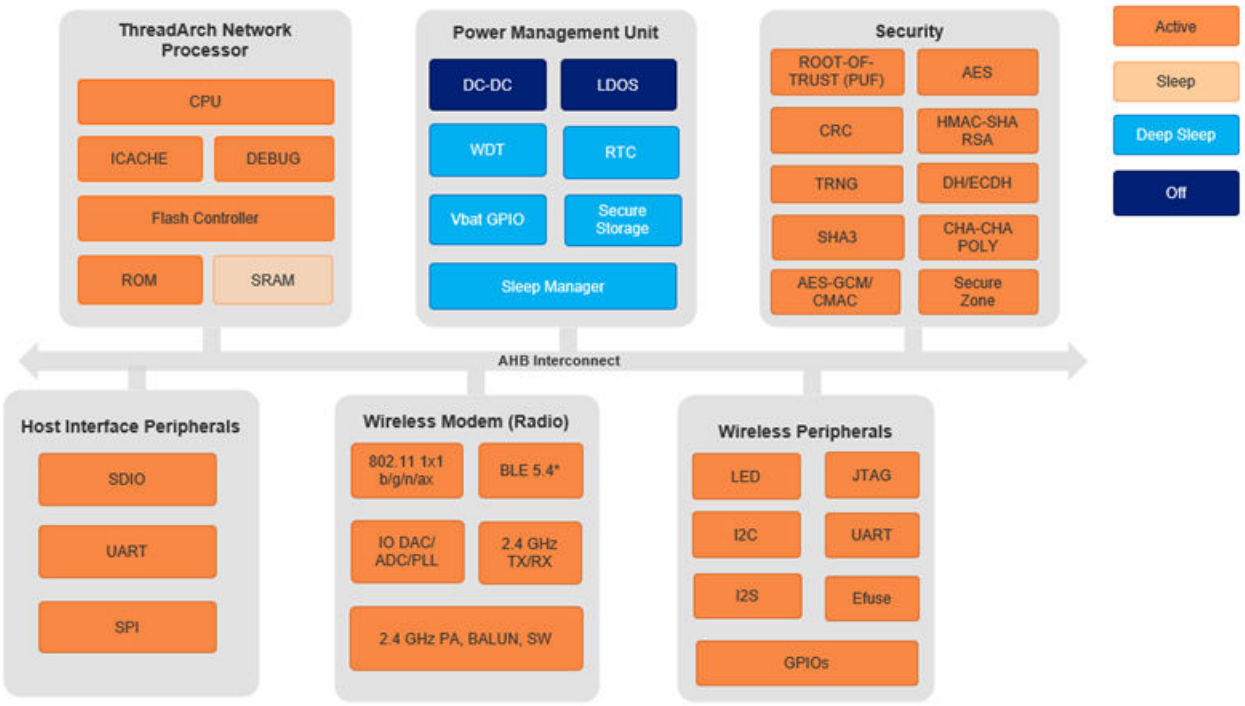

Figure 1.1. SiWx917 Power Domains

The power domains of SiWx917 in NCP mode are:

- **ThreadArch® Network Processor** (hereafter referred to as "NWP"): The multi-threaded processor that runs wireless and network stacks on independent threads.
- **Power Management Unit** (hereafter referred to as PMU): Responsible for supplying power required by various sections of SiWx917.
- **Security**: Contains security feature blocks like hardware accelerators, secure firmware update, etc.
- **Wireless Peripherals**: Contains various on-chip peripherals such as LED, UART, JTAG etc.,
- **Host Interface Peripherals**: Contains SDIO, UART, and SPI peripherals.

2. Power States

The SiWx917 can be in one of the following Power States:

- **Active state**: All the power domains are powered on.
- **Sleep state**: The PMU and SRAM domains are powered on, and the remaining power domains are powered off. The SRAM's contents are retained.
- **Deep Sleep state**: The PMU domain alone is powered on, and the remaining power domains are powered off. The SRAM's contents are not retained.
- **Off state**: Most sections in the PMU and the remaining power domains are powered off. More details about this state will be added in the next version.

The Operational Modes in Active State are explained in detail in the Active State Operational Modes section. The operational mode in Sleep state and Deep Sleep state are explained in detail in the Sleep State Operational Mode and Deep Sleep State Operational Mode sections respectively.

3. Active State Operational Modes

The SiWx917 can be in any of the three operational modes in its Active state. Each operational mode consumes a different amount of current.

1. Transmit Mode
2. Receive Mode
3. Listen Mode

3.1 Transmit Mode

In the Transmit Mode, all the power domains of the SiWx917 are powered on except the receiver sections of the Base Band Processor (BBP), Analog Front End (AFE), and RF Front End (RFFE) domains. This is the highest power-consuming mode.

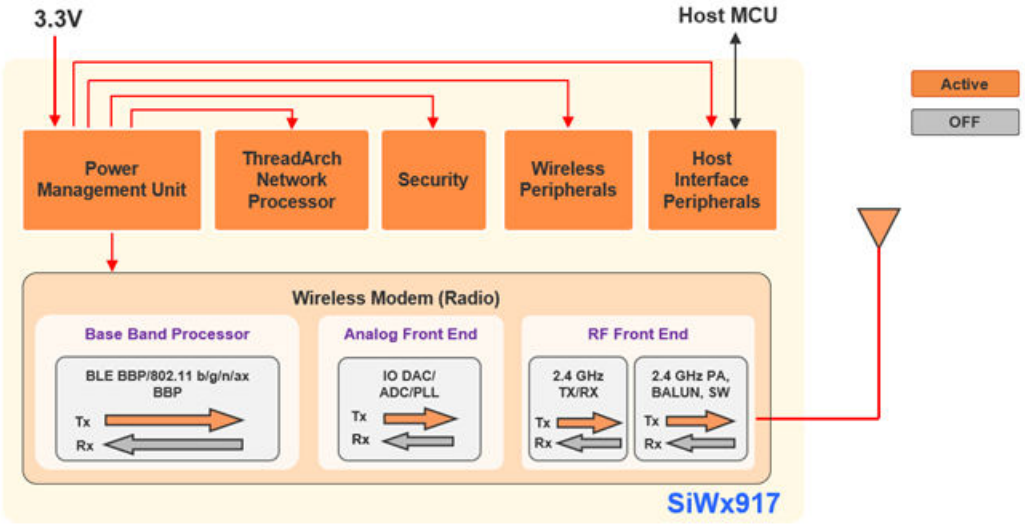

Figure 3.1. Transmit Mode in Active State

3.2 Receive Mode

In the Receive Mode, the transmit sections of the BBP, AFE, and RFFE are powered off.

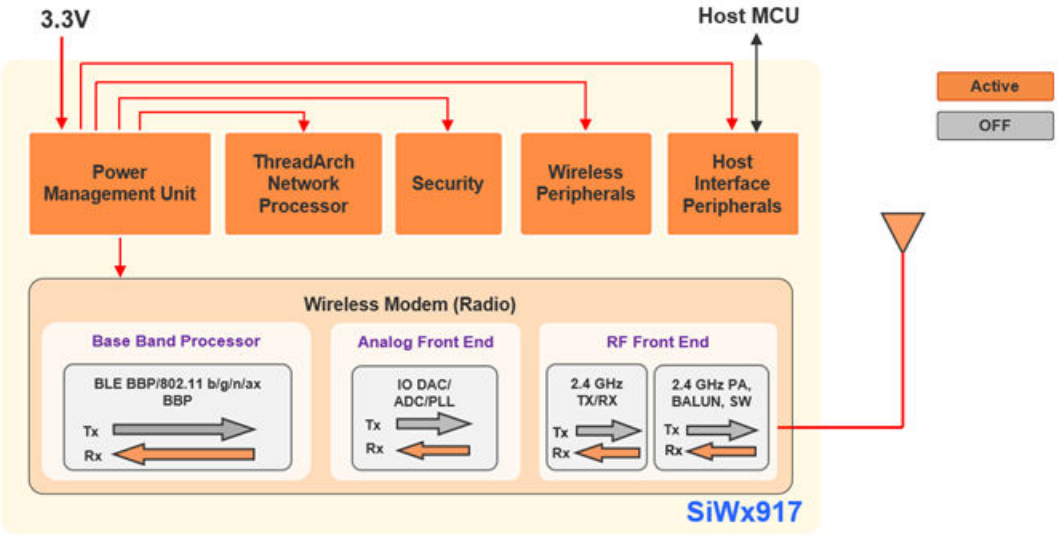

Figure 3.2. Receive Mode in Active State

3.3 Listen Mode

This mode is a subset of the receive mode. In Listen Mode, the transmit sections of BBP, AFE, and RFFE are powered off. Certain receive portions of the BBP and AFE are powered off as no packet reception is in progress.

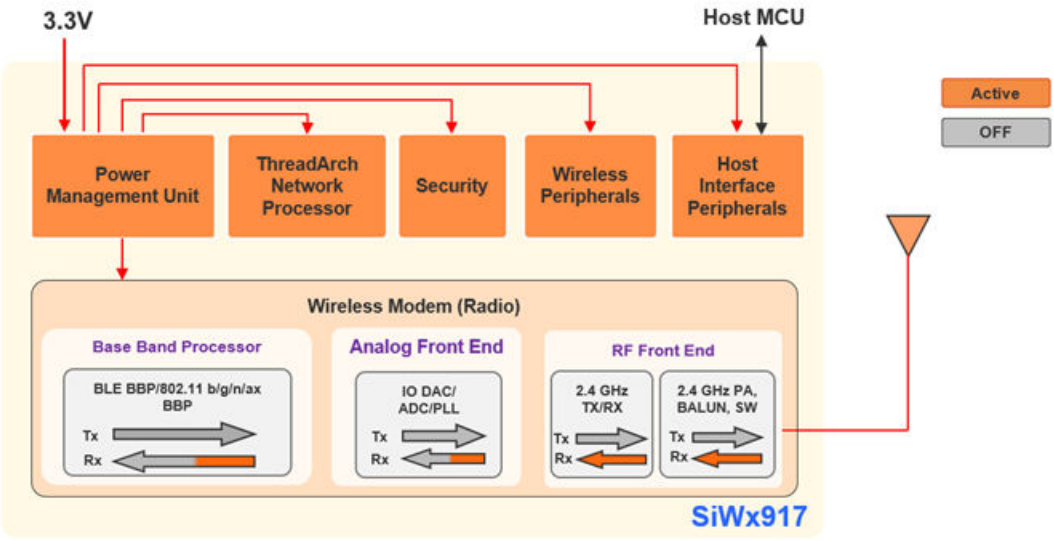

Figure 3.3. Listen Mode in Active State

4. Sleep State Operational Mode

The SiWx917's Sleep state Operational Mode is called **Ultra Low Power (ULP) Mode with RAM Retention**.

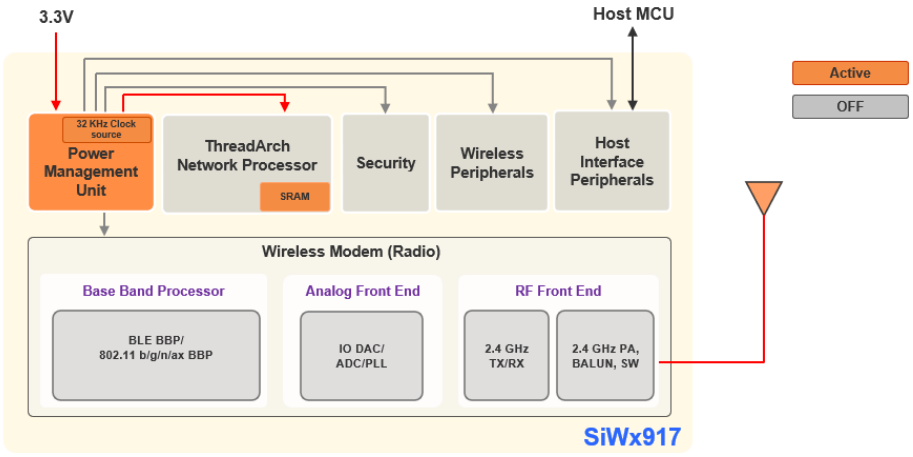

Figure 4.1. Ultra Low Power Mode with RAM Retention

- In the Ultra Low Power Mode with RAM Retention, the Wireless Modem, NWP, Security, Wireless Peripherals and Host Interface power domains are powered off. The SRAM contents and its current state are retained.
- The PMU has control over the other sections of the chip.
- The always-on logic domain operates on a lowered supply and a 32 KHz low-frequency clock to reduce power consumption.
- **Software Configuration**: The 32 KHz Clock Source required during ULP Mode is software configurable. The possible options are:
  1. Internal 32KHz RC clock (default)
  2. 32 KHz external crystal clock (enable BIT(22))
  3. 32 KHz bypass clock on UULP_VBAT_GPIO_3 (enable BIT(23))

Based on your choice, enable the respective bit in the boot configuration (sl_wifi_device_configuration_t.boot_config.ext_custom_feature_bit_map). For more details on 32 KHz Clock Source, refer to the **7.4.1 Clock Specification section** in the SiWx917 NCP Data Sheet and Chip Schematic.

- The SiWx917 NCP's Host Interface is inactive and thereby the interface is activated each time SiWx917 wakes up.

5. Deep Sleep State Operational Mode

The SiWx917's Deep Sleep state Operational Mode is called **Ultra Low Power Mode without RAM Retention**

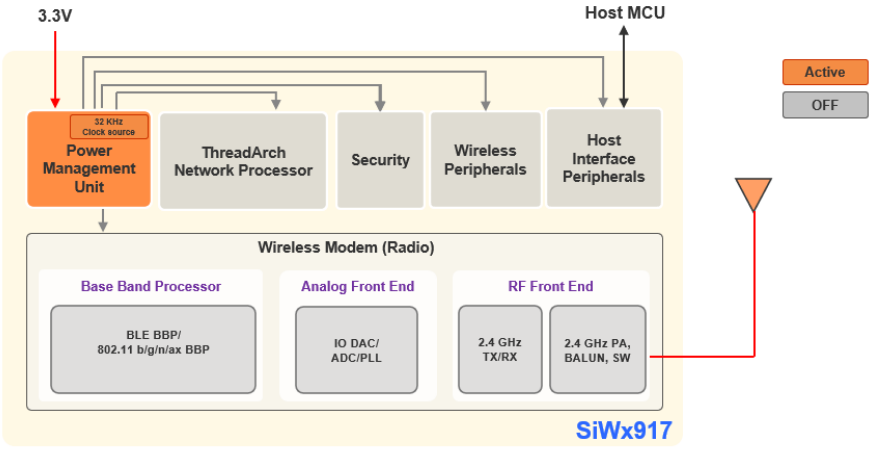

Figure 5.1. Ultra Low Power Mode without RAM Retention

- This mode differs from ULP Mode with RAM Retention with respect to SRAM's retention.
- The SRAM contents and current state are not retained.
- In this mode, the SiWx917 consumes the lowest possible current (~2.5 uA).

6. Power Modes

The SiWx917 can be in one of the following functional Power Modes or Performance Profiles:

- Active Mode or High-Performance Mode: The SiWx917 is always in Active state and data transmission/reception is immediate.
- Power Save Mode: The SiWx917 is in Sleep state mostly, consuming low power. In this mode, depending on the user application, the SiWx917 switches to:
  - Active State - for transmitting/receiving application data to/from the remote devices and processing host application commands
  - Sleep State – during idle state or inactivity

7. Power Save Modes

The Power Save Modes for SiWx917 in NCP mode are broadly classified into the following:

- Connected Power Save Modes
- Unconnected Power Save Modes

When the SiWx917 is set into Power Save Mode after a wireless connection (connect to Access Point via Wi-Fi or a BLE central device), it is said to be in Connected Power Save Mode. The Connected Power Save modes are classified into two types:

- Associated Power Save
- Associated Power Save with Low Latency

When the SiWx917 is set into Power Save Mode before establishing a wireless connection, it is said to be in Unconnected Power Save Mode. The Unconnected Power Save modes are classified into two types:

- Standby Power Save with RAM Retention
- Standby Power Save without RAM Retention

Before going through each of the Power Save Modes, the prerequisite is to understand the Power Save Handshake Mechanism between host and SiWx917 in its Power Save Mode.

Note:

1. For Connected Power Save Modes, ULP mode with RAM Retention configuration alone is supported, while for Unconnected Power Save Modes, ULP mode with and without RAM Retention configurations are supported.
2. The sleep/wake state switching in Power Save Modes can be via any of the selected handshake mechanisms mentioned in the Power Save Handshake Mechanism section.

7.1 Power Save Mode Handshake Mechanism

During its Power Save Mode, when the SiWx917 is in Sleep state, if the host application wants to send data/command to it, the host must wake the SiWx917 up. As the SiWx917's host Interface is powered off in Sleep state, a handshake mechanism between the host and SiWx917 is used for sleep/wake state switching. The handshake mechanism can be one of the following:

- General Purpose Input/Output (GPIO)-based
- Message-based

Note: While setting SiWx917 into Power Save Mode, no Handshake Mechanism is used. The Power Save Handshake Mechanism comes into the picture only when SiWx917 is in Power Save Mode.

7.1.1 GPIO-Based Handshake

The GPIO-based Handshake, as the name specifies uses GPIOs for handshake. During its Power Save Mode, the SiWx917 uses the following GPIOs for receiving sleep/wake requests from the host, and for sending its sleep/wake indications to the host.

- UULP_VBAT_GPIO_2 (Input to SiWx917): The host uses this GPIO for the following scenarios:
  - If the host application wants to send data/command to SiWx917, it makes a wake request by asserting this GPIO.
  - After communicating with SiWx917, the host lets the SiWx917 sleep by de-asserting this GPIO.
- UULP_VBAT_GPIO_0/UULP_VBAT_GPIO_3 (Input to host): The SiWx917 uses either of these GPIOs to send sleep/wake indications to the host.

Software Configuration:

- For making Ultra Low Power configuration with GPIO-based handshake, enable the SL_SI91X_FEAT_ULP_GPIO_BASED_HANDSHAKE feature in the Feature bit map (sl_wifi_device_configuration_t.boot_config.feature_bit_map).
- By default, the UULP_VBAT_GPIO_3 should be used by the host to receive sleep/wake indications from SiWx917. Should you want to use UULP_VBAT_GPIO_0 for receiving sleep/wake indications from SiWx917, enable SL_SI91X_FEAT_SLEEP_GPIO_SEL_BITMAP feature in config feature bit map (sl_wifi_device_configuration_t.boot_config.config_feature_bit_map).

Note: If UULP_VBAT_GPIO_3 is used for sleep/wake indications, do not choose 32 KHz bypass clock on UULP_VBAT_GPIO_3 (BIT(23)) for ULP operation. In this case, choose the internal 32 KHz RC clock or 32 KHz external crystal clock for ULP mode operation.

7.1.2 Message-based Handshake

If the host has a limited number of GPIOs available, the Message-based handshake can be used. The Message-based handshake, as the name specifies exchanges wake/sleep information through messages over the Host Interface for handshake.

- If the host application wants to send data/command to SiWx917, it needs to wait until it receives a "WKP" message from the host.
- Whenever the SiWx917 wakes up, it sends a "WKP" message to the host.
- Whenever the SiWx917 wants to enter sleep state, it requests the host by sending an "SLP" message.
- After receiving the "SLP" message from the SiWx917, the host, if done sending data/commands to SiWx917, gives sleep permission by sending an "ACK" message. The SiWx917 does not enter a sleep state until it receives an "ACK" message from the host.

Note: In Rev. 1.0 version of the document, the Message-based handshake is not supported. Refer to the Release Notes or Revision History of this document to check about Message-based handshake support.

7.2 Connected Power Save Modes

In Connected Power Save Modes, the SiWx917 enters Sleep state when idle and switches to active state for transmitting/receiving data to/from AP/host MCU. Before going through Connected Power Save Modes, it is important to understand the sleep/wake state switching mechanism and wake interval concepts of SiWx917 during Connected Power Save Mode.

7.2.1 Sleep/Wake State Switching Mechanism

1. The connecting station (here SiWx917) makes an Association request to the AP. In the Association frame, it sends the Listen Interval which indicates how often the station wakes to listen to the AP beacon frames. Based on the Listen Interval, the AP holds the data frames destined for the station.
2. The AP shall respond with an Association Response frame in which it specifies the Association Identifier (AID) to its connecting station.
3. The SiWx917 can be configured to wake every Delivery Traffic Indication Message (DTIM) Interval or Beacon Interval or Listen interval or Target Wake Time (TWT) Wake Interval during its Connected Power Save Mode.
   - Beacon Interval-based wakeup: Beacon Interval is the period between two subsequent beacon frames transmitted by AP. The station wakes every beacon interval.
   - DTIM Interval-based wakeup: The DTIM period specifies how often an AP beacon includes Buffered Traffic Indication to its connected clients via the TIM element in the beacon frame. When the AP includes TIM information in a beacon frame, the beacon is called a DTIM beacon. DTIM interval is the time between two subsequent DTIM beacons transmit-ted by AP. DTIM Interval = Beacon Interval*DTIM Period.
   - Listen Interval-based wakeup: Based on the listen interval configured in the application, the station wakes up at the nearest integral multiples of DTIM Interval/Beacon Interval broadcasted by the connected AP which is just less than or equal to the Listen Interval. This is explained in detail in the Listen Interval section.

     Note: Configuring larger listen intervals greater than 1000 ms might lead to AP disconnecting the SiWx917. It is highly recommended to use 1000 ms as a listen interval for low-power consumption. APs supporting Wi-Fi 6 offer longer sleep intervals beyond 1000 ms.

   - TWT Wake Interval-based wakeup: The station wakes every TWT Wake Interval configured in the application. The APs confirming Wi-Fi 6 support TWT. TWT is explained in detail in the Target Wake Time section.
4. When the station is set in Connected Power Save Mode by the host, it sends a Null data frame to the AP with the Power Management (PWR MGT) bit set in the Frame Control field, indicating that it is entering the Sleep State.
5. The AP acknowledges the Station's Null data frame.
6. When its connected stations are asleep,
   - if the AP has any broadcast/multicast frames to transmit, it buffers them. It indicates this information to its connected stations in the immediate DTIM beacon with the Multicast field in the Traffic Indication Map (TIM) set to "True" and transmits the broadcast/multicast frames to all its connected stations.
   - if the AP has unicast data frames for a station, it buffers the frames. The AP informs the station about the buffered data via the Partial Virtual Bitmap (PVB) field in the TIM element in the immediate DTIM beacon. The PVB determines the list of AIDs set. For example, if a station is assigned with AID "3" during its association, whenever there are data packets buffered for this station, the AP sets bit "3" in the PVB.

   Note: If the broadcast/multicast frames are not important for your application, call sl_wifi_filter_broadcast() API to ignore these frames before calling sl_wifi_set_performance_profile() API.

7. When the station wakes up to receive an AP beacon, it checks whether its AID is set in the Partial Virtual Bitmap (PVB) and if set retrieves the buffered frames from the AP.
8. The process of retrieving the unicast data from AP is different for different Connected Power Save Modes. The Associated Power Save Mode uses the Maximum Power Save Polling (Max PSP) Mechanism and Associated Power Save with Low Latency uses the Fast Power Save Polling (Fast PSP) mechanism. The Max PSP and Fast PSP mechanisms shall be explained in detail in the upcoming sections of the document.
9. After receiving the data frames, the station goes back to sleep state.

7.2.2 Beacon Interval

- The SiWx917 wakes every Beacon Interval (BI) configured in the AP. The more the Beacon Interval, the longer the SiWx917 stays asleep, thereby reducing the current consumption.
- The following figure illustrates the BI-based wakeup of SiWx917 for AP's BI = 100 ms and DTIM period =3. In this case, the SiWx917's wake interval = 100 ms.

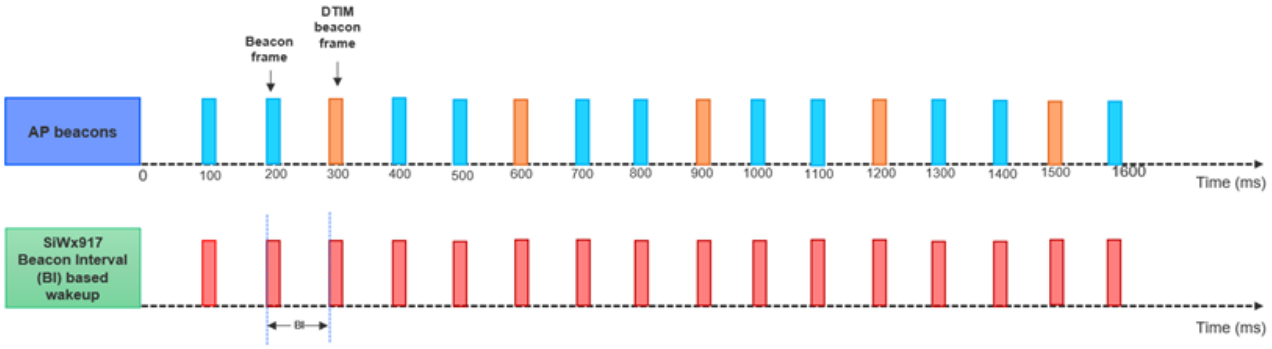

Figure 7.1. BI-based Wakeup

- **Configuration:** Set the structure variable sl_wifi_performance_profile_t.dtim_aligned_type to SL_SI91X_ALIGN_WITH_BEACON while calling sl_wifi_set_performance_profile() API.

7.2.3 DTIM Interval

- The SiWx917 wakes every DTIM Interval, as per the DTIM period configured in the AP. The less the DTIM Interval, the less the Rx latency to retrieve the buffered frames from AP.
- The following figure illustrates the DTIM Interval-based wakeup of SiWx917 for AP's BI = 100 ms and DTIM period = 3. In this case, the SiWx917's wake interval = 300 ms.

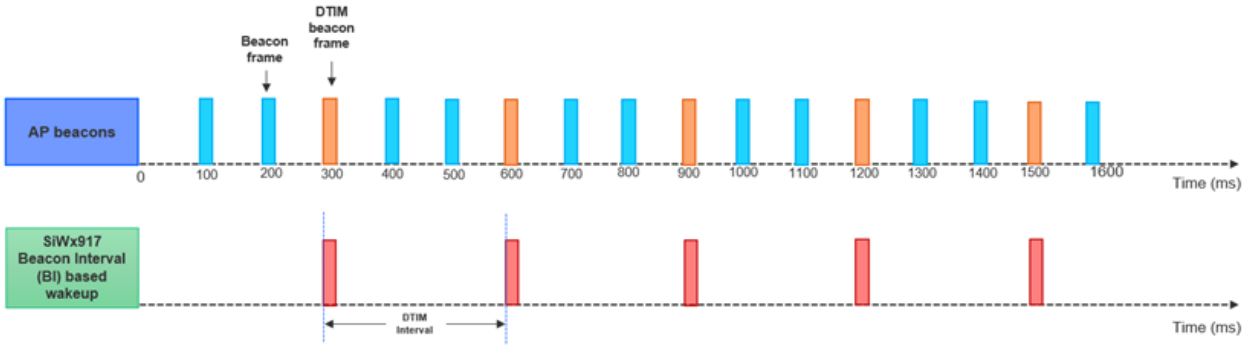

Figure 7.2. DTIM Interval-based Wakeup

- **Configuration:** Set the structure variable sl_wifi_performance_profile_t.dtim_aligned_type to SL_SI91X_ALIGN_WITH_DTIM_BEACON while calling sl_wifi_set_performance_profile() API.

7.2.4 Listen Interval

- The Listen Interval indicates how often the SiWx917 in Connected Power Save Mode wakes to listen to AP beacons.
- By default, the Listen Interval is set to 1000 ms in the WiSeConnect SDK v3.x.
- The wake interval of SiWx917 can be configured to be aligned with DTIM Interval or Beacon Interval of AP.
- For example, if the Listen Interval is 1000 ms, the DTIM period is 3 and beacon interval is 100 ms at AP (it means every 3rd beacon is a DTIM beacon),
  - the SiWx917 wakes every 900 ms (<= Listen Interval) if configured to be aligned with DTIM period of AP (DTIM Interval-based wakeup).
  - the SiWx917 wakes every 1000 ms (<= Listen Interval) if configured to be aligned with beacon interval of AP (Beacon Interval-based wakeup). In this case, the SiWx917 does not take DTIM period of AP into consideration.
- **Configuration**: The Listen Interval can be configured in two methods:
  - During association with an AP.
    - Set the listen_interval to a desired value in multiples of DTIM period of AP and call the following API. void sl_si91x_set_listen_interval(uint32_t listen_interval)
    - Set the join_feature_bitmap to SI91X_JOIN_FEAT_LISTEN_INTERVAL_VALID and call this API before calling Wi-Fi connection APIs: sl_status_t sl_si91x_set_join_configuration(sl_wifi_interface_t interface, uint8_t join_feature_bitmap)
    - Call sl_net_up() API.
  - After establishing a Wi-Fi connection with the AP, pass the listen interval as an argument for power save API.
    - Set the join_feature_bitmap to SI91X_JOIN_FEAT_PS_CMD_LISTEN_INTERVAL_VALID.
    - Call the sl_net_up() for associating with an AP.
    - Call the following API with the profile structure variable's member (sl_wifi_performance_profile_t.listen_interval) set to a desired listen interval. This listen interval should be less than listen interval set during association request: sl_status_t sl_wifi_set_performance_profile(const sl_wifi_performance_profile_t *profile)

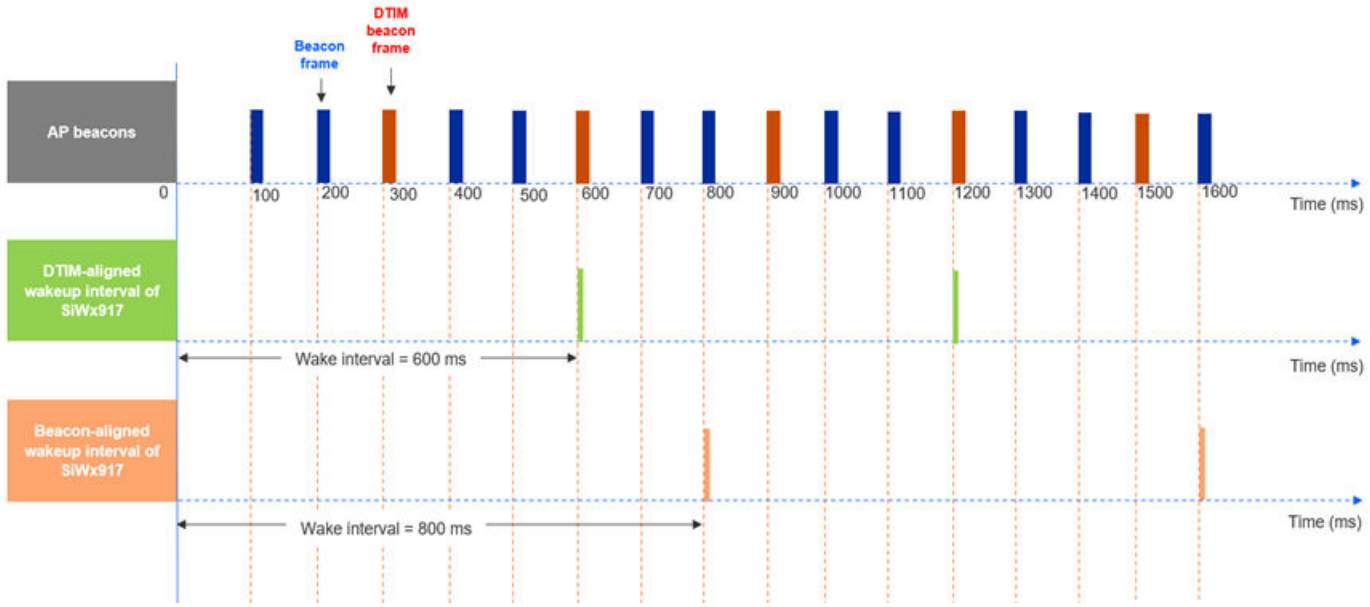

Figure 7.3. Listen Interval-based Wakeup

- The following figure illustrates the LI-based wakeup of SiWx917 for AP's BI = 100 ms and DTIM period = 3, LI = 800 ms. In this case, the SiWx917's wake interval = 600 ms with DTIM-aligned configuration and 800 ms with Beacon-aligned configuration.

Note: For Listen Interval based wake up, the broadcast and multicast frames transmitted by the AP when the device is asleep may be lost.

7.2.5 Associated Power Save Mode

The Associated Power Save Mode is a combination of Max PSP Mode with GPIO-based handshake between the host and SiWx917.

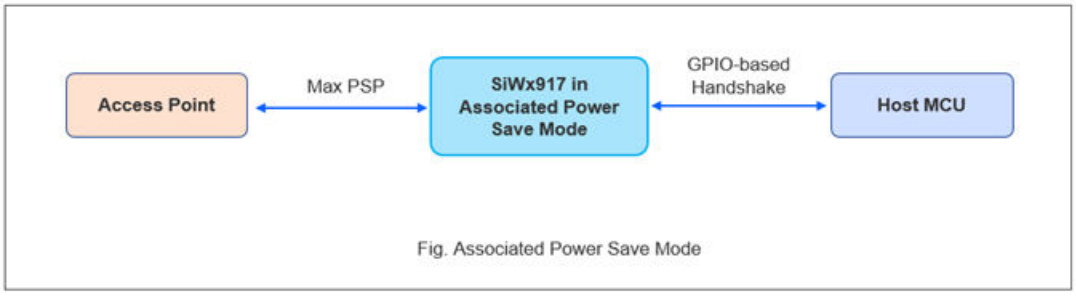

Figure 7.4. Associated Power Save Mode

Communication between AP and SiWx917:

The SiWx917, when in Associated Power Save Mode, can send data to AP at any instance. For retrieving unicast data buffered at the AP, the Max PSP mechanism is used.

Maximum Power Save Polling (Max PSP):

- Whenever the AP receives data frames that are destined for a station (here SiWx917), it buffers the frames.
- The AP informs this to the station by setting the corresponding station's AID in the TIM element of the next immediate DTIM beacon.
- On the next wake-up (based on DTIM Interval of BI or listen interval), the Station receives the beacon and checks for the TIM.
- If the AID of the station is set, it sends a Power Save Polling (PS-Poll) frame with its AID to the AP to retrieve a data frame.
- The AP acknowledges the PS-Poll frame and transmits a data frame with the "More Data" field set to 1 in case there are more data frames buffered for the station.
- After receiving a data frame, the SiWx917 station sends it to the host.
- The station sends a PS-Poll frame to retrieve each data frame from the AP.
- While sending the last data frame to the station, the AP shall set the "More data" field to "0".
- After receiving the last data frame, the station goes back to sleep state.

The Max PSP saves more power but produces lower throughputs. In case of receiving bulk amount of data, sending a PS-Poll frame to retrieve each frame affects the throughput and induces a considerable amount of delay.

Communication from SiWx917 to the host:

- Whenever the SiWx917 wants to send data to the host, no handshake is required.
- During Connected Power Save Mode, the SiWx917 acknowledges its sleep/wake states to the host. Whenever the SiWx917 wakes up for every DTIM/Beacon/Listen Interval, it asserts the UULP_VBAT_GPIO_0/UULP_VBAT_GPIO_3. After receiving the AP beacon, if the TIM element of the beacon indicates no data buffered to it, the SiWx917 goes back to sleep by de-asserting the UULP_VBAT_GPIO_0/UULP_VBAT_GPIO_3.

Communication from the host to SiWx917:

- Whenever the host wants to send data/commands to SiWx917, a GPIO-based handshake is required. The host makes a wake request to SiWx917 by asserting the UULP_VBAT_GPIO_2. The host must wait until it receives a wake indication from SiWx917.
- After SiWx917 wakes up from Sleep state, it gives a wake indication to the host by de-asserting the UULP_VBAT_GPIO_0/ UULP_VBAT_GPIO_3.
- The host, after sending the data/commands to SiWx917, gives sleep permission to SiWx917 by de-asserting the UULP_VBAT_GPIO_2.
- The SiWx917 acknowledges the host's sleep permission by de-asserting the UULP_VBAT_GPIO_0/UULP_VBAT_GPIO_3 and goes back to sleep state.

The following flowchart illustrates the Associated Power Save Mode:

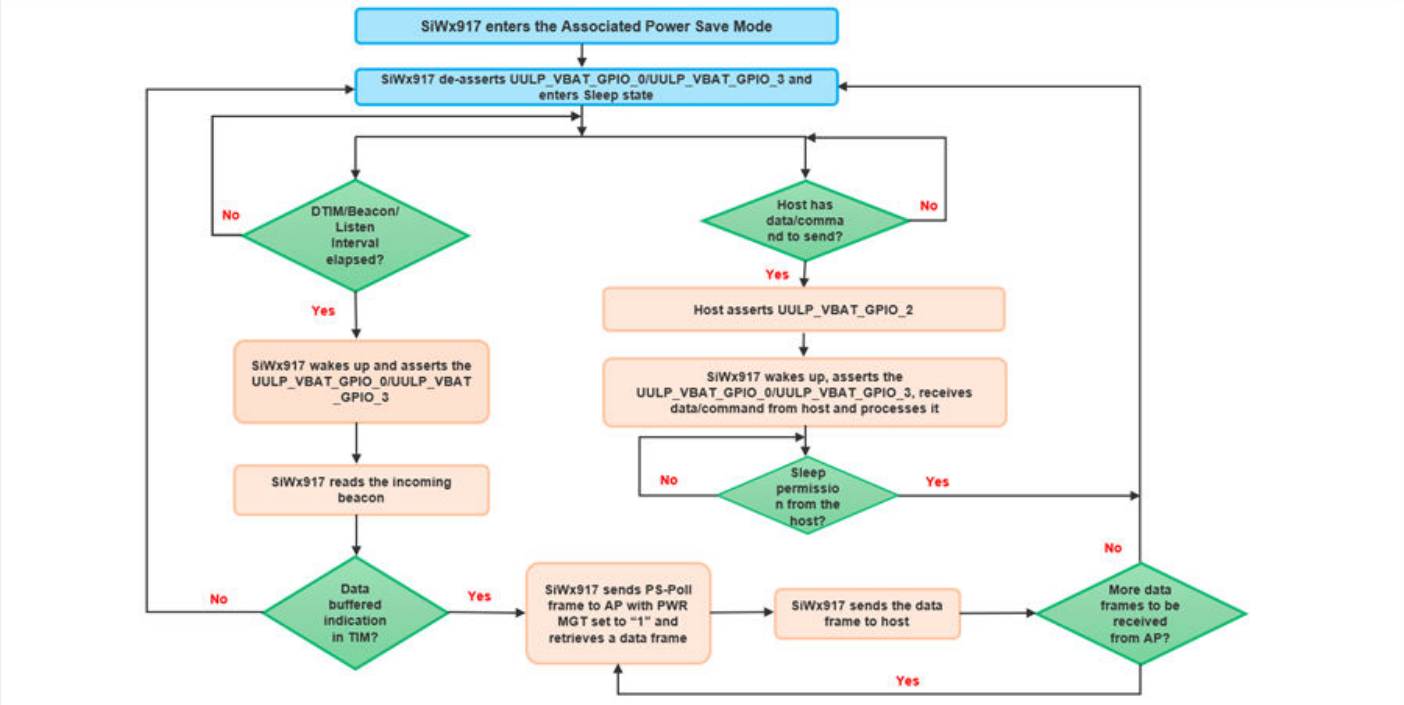

Figure 7.5. Associated Power Save Mode Flowchart

Configuration:

1. After a wireless connection, the SiWx917 can be set into Associated Power Save by calling the below API with the profile structure variable's member (sl_wifi_performance_profile_t.sl_performance_profile_t) set to ASSOCIATED_POWER_SAVE.

   sl_status_t sl_wifi_set_performance_profile(const sl_wifi_performance_profile_t *profile)
2. The listen interval can be configured as described in the Sleep/Wake States Switching Mechanism section.

7.2.6 Associated Power Save with Low Latency Mode

The Associated Power Save with Low Latency Mode is a combination of Fast PSP Mode with GPIO-based handshake between the host and SiWx917.

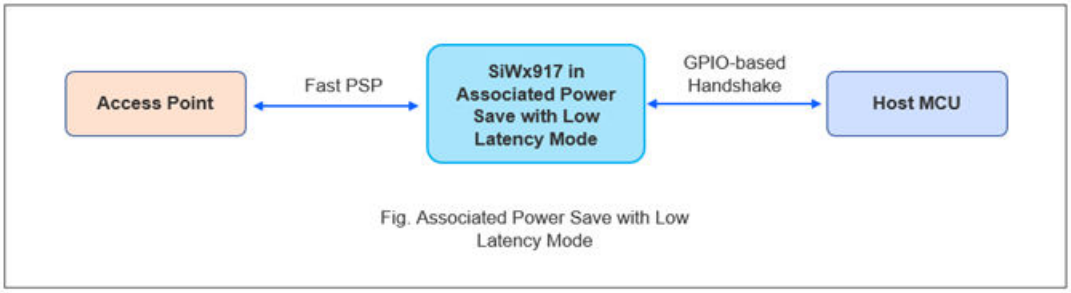

Figure 7.6. Associated Power Save with Low Latency Mode

Communication between AP and SiWx917:

The SiWx917, when in the associated Power Save Mode with Low Latency, can send data to AP at any instance. For retrieving unicast data buffered at the AP, the Fast PSP mechanism is used.

Fast Power Save Polling (Fast PSP):

- Whenever the AP receives data frames that are destined for a station (here SiWx917), it buffers the frames.
- The AP informs this to the station by setting the corresponding station's AID in the TIM element of the next immediate DTIM beacon.
- On the next wake-up (based on DTIM or listen interval), the station receives the beacon and checks for the TIM.
- If the AID of the Station is set, the station exits power save mode, switches to active mode, and sends a Null data frame with PWR MGT bit set to "0" to the AP to retrieve all the data packets from AP.
- The AP acknowledges the Null data frame and it AP transmits a data frame with the "More Data" field set to 1 in case there are more data frames buffered for the Station.
- After receiving a data frame, the SiWx917 station sends it to the host.
- After receiving the last data frame, the station waits for the monitor interval time configured and checks for data frames from AP.
- If no data is to be received, the station goes back to sleep state by sending a Null data frame with the PWR MGT bit set to "1" to the AP.

The Fast PSP saves less power and produces better throughput when compared to the Max PSP. If both throughput and power save are important for your application, use Fast PSP.

Communication from SiWx917 to the host:

The procedure is the same as mentioned in the Communication from SiWx917 to host subsection of Associated Power Save Mode section.

Communication from the host to SiWx917:

The procedure is the same as mentioned in the Communication from SiWx917 to host subsection of Associated Power Save Mode section.

The following flowchart illustrates the Associated Power Save with Low Latency Mode:

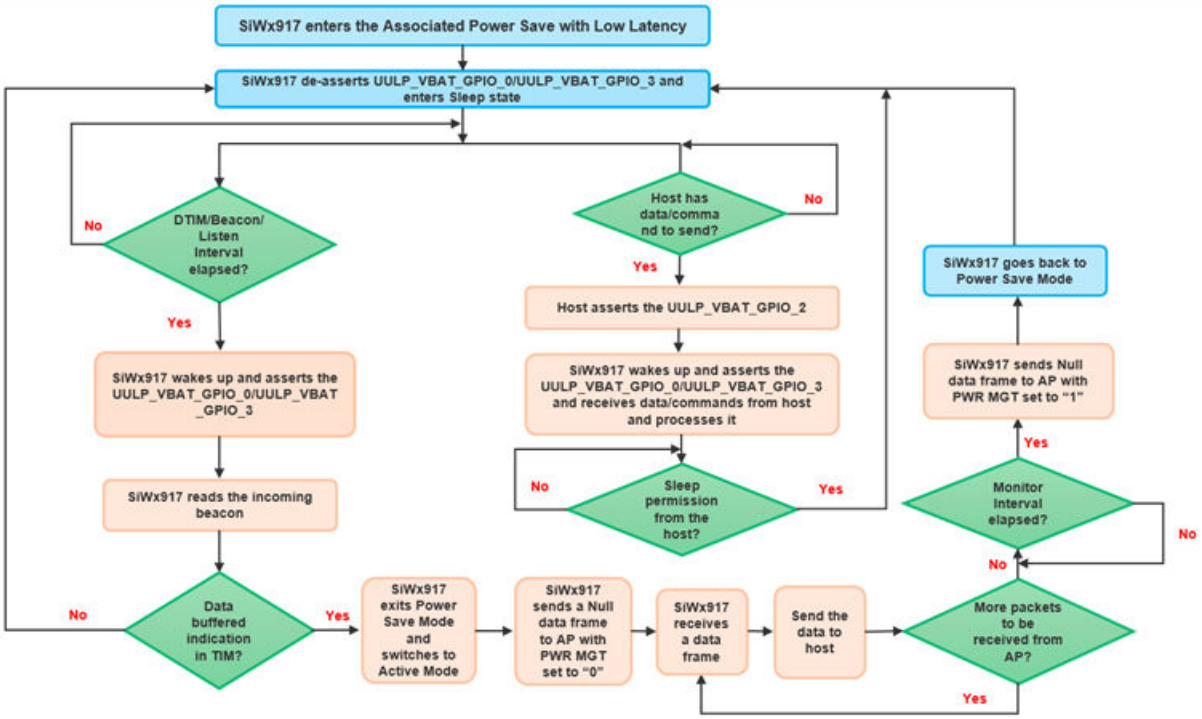

Figure 7.7. Associated Power Save with Low Latency Mode Flowchart

Configuration:

1. After a wireless connection, the SiWx917 can be set into Associated Power Save with Low Latency by calling the below API with the profile structure variable's member (sl_wifi_performance_profile_t.sl_performance_profile_t) set to ASSOCIATED_POWER_SAVE_LOW_LATENCY. The Monitor Interval can be configured by defining the profile structure variable's member (sl_wifi_performance_profile_t.monitor_interval).

   sl_status_t sl_wifi_set_performance_profile(const sl_wifi_performance_profile_t *profile)
2. The Listen Interval can be configured as described Sleep/Wake States Switching Mechanism subsection.

7.2.7 Enhanced Max PSP Feature

In Associated Power Save Mode, during unicast data retrieval, some Access Points do not deliver buffered data destined for the station for a PS-Poll frame. This interoperability issue can be avoided with the Enhanced Max PSP feature.

Functionality and Usage:

- Initially, the SiWx917 is in Associated Power Save Mode.
- After sending a PS-Poll frame to AP, if the AP delivers the buffered data within 20ms, the SiWx917 continues to be in Associated Power Save Mode.
- When the AP does acknowledge PS-Poll and does not deliver the buffered data within 20ms, the SiWx917 switches to Associated Power Save with Low Latency Mode.
- To use this feature, do the following configuration:
- Enable the SL_SI91X_ENABLE_ENHANCED_MAX_PSP in config_feature_bit_map (sl_wifi_device_configuration_t.boot_config.config_feature_bit_map).
- Set the Performance Profile to ASSOCIATED_POWER_SAVE_LOW_LATENCY.

Note: To switch the SiWx917 from connected power save to active mode, call the sl_wifi_set_performance_profile() API with the profile structure variable's member (sl_wifi_performance_profile_t.sl_performance_profile_t) set to HIGH_PERFORMANCE.

7.2.8 Target Wake Time (TWT)

TWT is a beneficial Wi-Fi 6 feature which allows connected stations to manage power efficiently with the reduced network contention.

- Reduces network contention: In the Legacy Power Save Modes, the Wi-Fi stations go to sleep and wake up at random times to perform data transfer without knowing the wake-up timings of other stations. With TWT, the AP schedules wake timings for its connected stations, ensuring no two Wi-Fi stations wake at the same time. This method helps avoid packet collisions, thus reducing retransmissions and in turn reducing the station's current consumption.

There are two types of TWT:

- Individual TWT: The connected stations negotiate the TWT session parameters (TWT Wake interval and Wake duration) independently with the AP.
- Broadcast TWT: AP broadcasts predefined TWT parameters (TWT Wake Interval and Wake duration) in its beacon and schedules different wake times for different connected stations.

The Wake duration, Wake Interval, and Sleep duration are derived from the TWT parameters.

- Wake duration: The duration for which the station is awake to transmit/receive data frames to/from the AP. This duration is also called as TWT Service Period (SP).
- Wake Interval: The time interval between two successive SPs' start times.
- Sleep duration: The time between the end of current SP and the start of next/future SP.

Configuration:

- Configure the TWT parameters using the sl_wifi_performance_profile_t. sl_wifi_twt_request_t structure.

```
typedef struct {
  uint8_t wake_duration;
  uint8_t wake_duration_tol;
  uint8_t wake_int_exp;
  uint8_t wake_int_exp_tol;
  uint16_t wake_int_mantissa;
  uint16_t wake_int_mantissa_tol;
  uint8_t implicit_twt;
  uint8_t un_announced_twt;
  uint8_t triggered_twt;
  uint8_t negotiation_type;
  uint8_t twt_channel
  uint8_t twt_protection;
  uint8_t twt_flow_id
  uint8_t restrict_tx_outside_tsp;
  uint8_t twt_retry_limit
  uint8_t twt_retry_interval;
  uint8_t req_type
  uint8_t twt_enable
  uint8_t wake_duration_unit;
} sl_wifi_twt_request_t;
```

Note: To know more about the above structure and its configuration, refer to TWT TCP Client example in the WiSeConnect SDK v3.x.

- As per above structure parameters, below are the derivations:
  - The Wake duration or TWT Service Period is calculated as wake_duration*wake_duration_unit.
  - The TWT Wake Interval is calculated as (wake_int_mantissa)*(2^wake_int_exp).
  - The TWT Sleep duration is calculated as TWT Wake Interval - Wake duration.
- Define a sl_wifi_performance_profile_t. sl_wifi_twt_request_t structure variable and pass it as an argument to the below API.

  sl_status_t **sl_wifi_enable_target_wake_time**(sl_wifi_twt_request_t *twt_req)

7.2.9 TWT-based Power Save Mechanism

This section explains the TWT Power Saving Mechanism of SiWx917 with following TWT parameters set:

- Individual TWT - the connected station (SiWx917) initiates a negotiation with its individual TWT Wake interval and Wake duration.
- Implicit TWT - the TWT requesting STA (SiWx917) calculates the next TWT by adding a fixed value to the current TWT value.
- Unannounced TWT - the TWT requesting STA (SiWx917) does not announce its wake-up to AP through PS-POLLs or UAPSD Trigger frames.
- Non-Triggered TWT - The TWT SP does not contain any triggered frames.

Communication between AP and SiWx917:

- The SiWx917 sends an Association frame with TWT Requester Support bit set in the High Efficiency (HE) MAC capabilities field.
- After connecting to an Access Point, the SiWx917 transmits a TWT Setup frame with the TWT parameters depending on the request type (Request TWT/Suggest TWT/Demand TWT).
- If the AP agrees on the TWT parameters, it transmits a TWT Accept frame.
- The SiWx917 sends a Null data frame and goes to sleep.
- Whenever the AP receives destined packets for a station (SiWx917), it buffers them.
- The AP indicates this to the station by setting the corresponding AID in the TIM element of the next beacon.
- The SiWx917 wakes just before its TWT to read the incoming beacon from the AP for Time Synchronization.
- As soon as it reads the beacon, the SiWx917 goes back to sleep and wakes at start of TWT SP.
- During its Wake duration/TWT SP, if the AID of SiWx917 is set in the TIM element, the SiWx917 exits power save mode, switches to active mode, and retrieves all the data packets from AP during TWT SP.
- The AP transmits a data frame with the "More Data" field set to 1 in case there are more data packets buffered for the Station.
- After receiving a data packet, the SiWx917 sends it to the host.
- In case there are no data frames to be received, the station will be active for the rest of wake duration (if remains) and goes back to Sleep state.
- After TWT SP, if there are more data frames to be transmitted or received from the AP, the device still goes to sleep and shall carry out these data transfers at its next TWT SP.

TWT Mechanism can be used when your host application has predictable and deterministic data traffic. The host application should be confident about the times at which data traffic is expected should be known prior.

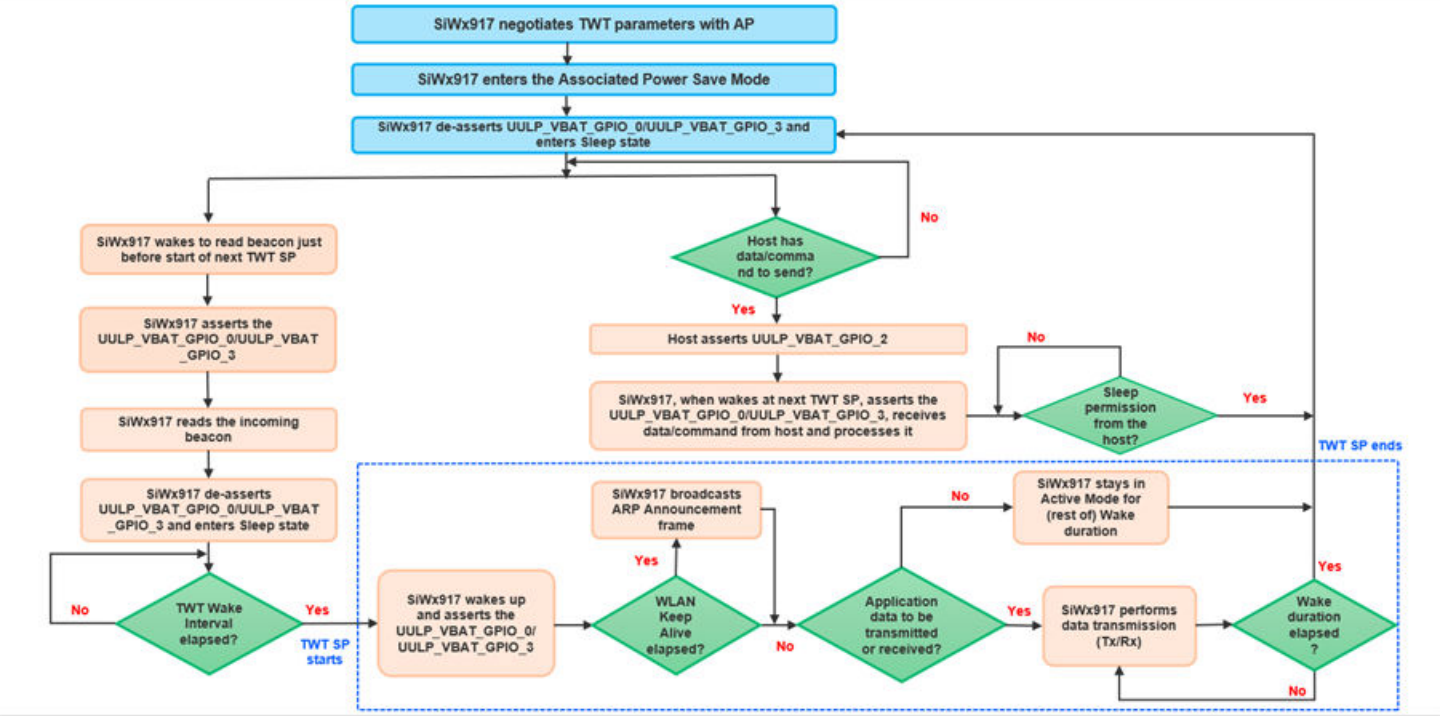

Figure 7.8. TWT Power Saving Mechanism

Communication from SiWx917 to the host:

- Whenever the SiWx917 wants to send data to the host, no handshake is required.

- The SiWx917 acknowledges its sleep/wake states to the host. Whenever the SiWx917 wakes every TWT Wake Interval, it as-serts the UULP_VBAT_GPIO_0/UULP_VBAT_GPIO_3. After TWT SP, the SiWx917 goes back to sleep by de-asserting the UULP_VBAT_GPIO_0/UULP_VBAT_GPIO_3.

Communication from the host to SiWx917:

- Whenever the host wants to send data/commands to SiWx917, a GPIO-based handshake is required. The host makes a wake request to SiWx917 by asserting the UULP_VBAT_GPIO_2. The host must wait until it receives a wake indication from SiWx917.
- The SiWx917 shall give a wake indication only when it wakes at next TWT SP by de-asserting the UULP_VBAT_GPIO_0/ UULP_VBAT_GPIO_3.
- The host can communicate with SiWx917 only when SiWx917 is awake for serving TWT SP. The host, after sending the data/ command to SiWx917, gives sleep permission to SiWx917 by de-asserting the UULP_VBAT_GPIO_2.
- If the host has more data/commands to send beyond TWT SP/SiWx917's Wake duration, it must wait until next TWT SP.

Configuration:

- After calling sl_net_up() API, register a callback function for TWT negotiation response events using the below API:

  static inline sl_status_t **sl_wifi_set_twt_config_callback**(sl_wifi_twt_config_callback_t function, **void** *optional_arg)
- Trigger the TWT parameters negotiation by calling the below API with the profile structure variable member sl_wifi_performance_profile_t.sl_wifi_twt_request_t set with TWT setup configuration.

  sl_status_t **sl_wifi_enable_target_wake_time**(sl_wifi_twt_request_t *twt_req)
- Once the AP sends a TWT response frame, the registered callback function gets triggered. The callback function gives the agreed TWT parameters during TWT parameters negotiation.
- Next, set the SiWx917 into Associated Power Save (or with Low Latency) by calling the below API with the profile structure variable's member (sl_wifi_performance_profile_t.sl_performance_profile_t) set to ASSOCIATED_POWER_SAVE (or ASSOCIATED_POWER_SAVE_LOW_LATENCY).

  sl_status_t sl_wifi_set_performance_profile(const sl_wifi_performance_profile_t *profile)
- To tear down the TWT session, set thesl_wifi_twt_request_t. twt_enable to "0" and call the below API.

  sl_status_t sl_wifi_disable_target_wake_time(sl_wifi_twt_request_t *twt_req)

7.2.10 Automatic TWT

Automatic TWT (Auto TWT) is a Silicon Labs's implementation that configures TWT parameters automatically based on the user application requirements. The user can provide the application requirements in terms of average Throughput and Rx latency. Based on these inputs, the SiWx917 automatically configures the TWT parameters and negotiates them with the AP.

Working and Configuration:

- After calling sl_net_up() API, register a callback function for TWT negotiation response events using the below API:

  static inline sl_status_t **sl_wifi_set_twt_config_callback**(sl_wifi_twt_config_callback_t function, **void** *optional_arg)
- The Auto TWT selection parameters are configured using the sl_wifi_performance_profile_t. sl_wifi_twt_selection_t structure.

```
typedef struct {
  uint8_t twt_enable;
  uint16_t average_tx_throughput;
  uint32_t tx_latency;
  uint32_t rx_latency;
  uint16_t device_average_throughput;
  uint8_t estimated_extra_wake_duration_percent;
  uint8_t twt_tolerable_deviation;
  uint32_t default_wake_interval_ms
  uint32_t default_minimum_wake_duration_ms;
  uint8_t beacon_wake_up_count_after_sp;
} sl_wifi_twt_selection_t;
```

- From the above structure, the configurable values are:
  - twt_enable : 1- Setup ; 0 - teardown
  - rx_latency : The maximum allowed receive latency, in milliseconds, when an Rx packet is buffered at the AP. If rx_latency is less than <= 2 sec, session creation is not possible. If configured as 0, then by default 2sec is considered.
  - avg_tx_throughput : The expected average Tx throughput in Kbps should be between 0 and half of device average throughput.
- Define a sl_wifi_performance_profile_t. sl_wifi_twt_selection_t structure variable and pass it as an argument to the below API.

  sl_status_t **sl_wifi_target_wake_time_auto_selection**(sl_wifi_twt_selection_t *twt_auto_request)
- When the above API is called, the SiWx917 automatically configures the TWT parameters and triggers a TWT negotiation with the AP.
- Once the AP sends a TWT response frame, the registered callback function gets triggered. The callback function gives the agreed TWT parameters during TWT parameters negotiation.
- Next, set the SiWx917 into Associated Power Save (or with Low Latency) by calling the below API with the profile structure variable's member (sl_wifi_performance_profile_t.sl_performance_profile_t) set to ASSOCIATED_POWER_SAVE (or ASSOCIATED_POWER_SAVE_LOW_LATENCY).

  sl_status_t sl_wifi_set_performance_profile(const sl_wifi_performance_profile_t *profile)
- The SiWx917 performs Tx/Rx based on the user configured parameters ensuring low latency for transmitting and receiving data.

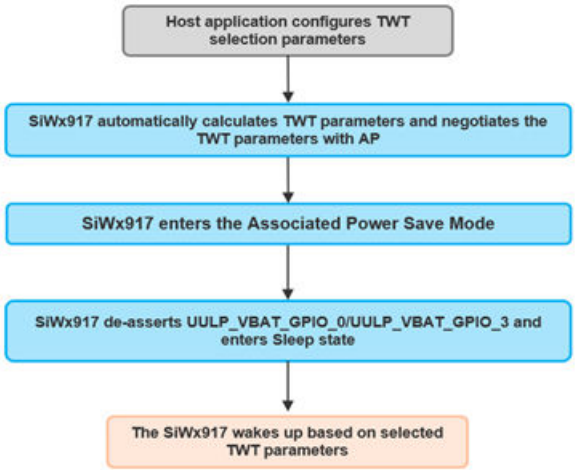

Figure 7.9. Auto TWT Power Save Mechanism

7.2.11 Benefits of Automatic TWT Over Standard TWT

- Power Optimization when compared to standard TWT.
- Low Latency for transmitting and receiving application data.

7.3 Unconnected Power Save Modes

In Unconnected Power Save Mode, the SiWx917 enters:

- Sleep state when idle and switches to active state for communicating with host MCU.
- Deep Sleep state when idle and exits Power Save Mode for communicating with host MCU.

7.3.1 Standby Power Save with RAM Retention Mode

- The Standby Power Save with RAM Retention Mode uses a GPIO-based handshake between the host and SiWx917.
- The SiWx917's RAM contents and current state are retained.
- The host should not send any wireless commands (scan for Wi-Fi networks, connect to a Wi-Fi network) to SiWx917 when configured in this mode.
- To perform scan or connect to a Wireless network, the SiWx917 should exit Power Save Mode. To achieve this, the host should call sl_wifi_set_performance_profile() with sl_wifi_performance_profile_t.sl_performance_profile_t set to HIGH_PERFORMANCE to make SiWx917 exit power save mode and switch to active mode and then send the wireless commands.

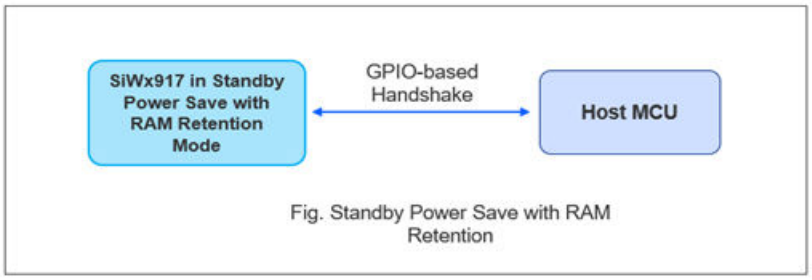

Figure 7.10. Standby Power Save with RAM Retention Mode

Communication from SiWx917 to host:

When the SiWx917 wants to send a command response to the host, no handshake is required. The SiWx917 shall not send any asynchronous data to the host.

Communication from host to SiWx917:

- Whenever the host wants to send a command to SiWx917, a GPIO-based handshake is required.
- The host makes a wakeup request to SiWx917 by asserting the ULP_GPIO_5 (in Low Power Mode)/UULP_VBAT_GPIO_2 (in Ultra Low Power Mode). The host must wait until it receives a wake indication from SiWx917.
- The host, after sending the command to SiWx917, gives sleep permission to SiWx917 by de-asserting the UULP_VBAT_GPIO_0/ UULP_VBAT_GPIO_3.
- The SiWx917 processes the received command, responds to the host, acknowledges the host's sleep permission by de-asserting the UULP_VBAT_GPIO_0/UULP_VBAT_GPIO_3 and goes back to sleep state.

The following flowchart illustrates the Standby Power Save with RAM Retention Mode:

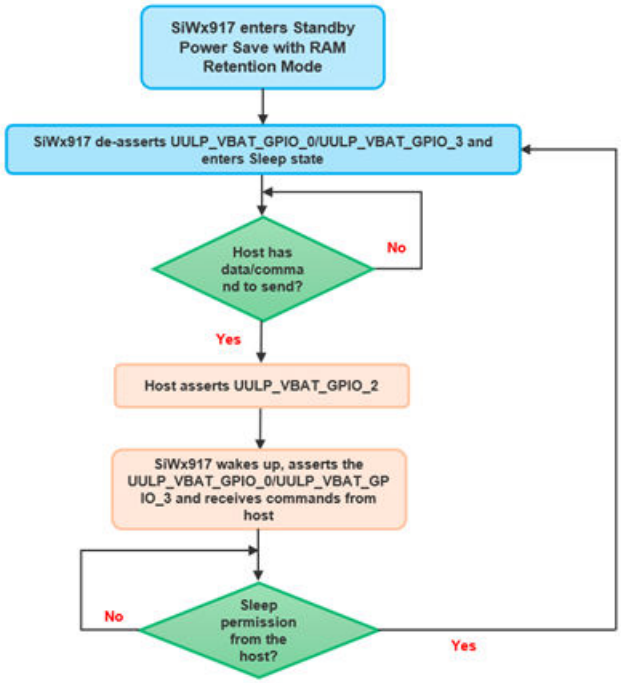

Figure 7.11. Standby Power Save with RAM Retention Mode Flowchart

7.3.2 Standby Power Save Mode

- The Standby Power Save Mode uses a GPIO-based handshake between the host and SiWx917.
- The SiWx917's RAM contents and current state are not retained.
- The host should not send any commands to SiWx917, when in Standby Power Save Mode.

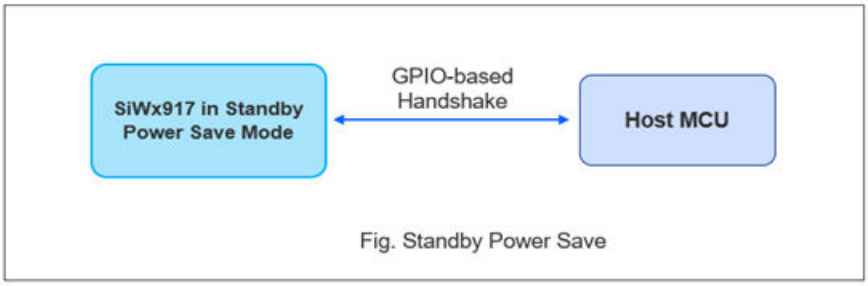

Figure 7.12. Standby Power Save without RAM Retention Mode

Communication from SiWx917 to host:

The SiWx917 shall not send any asynchronous data to the host when in Standby Power Save Mode.

Communication from host to SiWx917:

To send data/commands to SiWx917, the host should bring the SiWx917 back to active mode. For this, the host needs to re-initialize the SiWx917 as the previous state of the device is not retained.

Configuration:

- After the device initialization and before establishing a wireless connection, the SiWx917 can be set into Standby Power Save Mode by calling the below API with the profile structure variable's member (sl_wifi_performance_profile_t.sl_performance_profile_t) to STANDBY_POWER_SAVE.
- To switch the SiWx917 from Standby Power Save mode to active mode, the host needs to call si91x_req_wakeup() API and re-initialize the SiWx917 by calling the sl_net_init() API to initialize the device and bring it back to active mode.

7.4 How to Choose Power Save Modes?

This section describes how to choose a right Power Save Mode depending on the application and use case. Provided below are some typical use cases and Power Save Mode to be used in those use cases.

1. Let us say for an application, Rx latency is critical, and the data must be retrieved within one second. In such cases, choose Associated Power Save Mode with a listen interval of 1 second. If data retrieval should happen in less than a second, choose DTIM Interval or Beacon Interval based wake up without listen interval configuration.
2. Consider an application which,
   - performs periodic sensor data publishes to the cloud every 30 seconds.
   - upon a user request, performs instant download of heavy content updates/files of say > 500 KB size from a Cloud Server 15-20 times a day with low latency (up to 1 second to initiate a download).

   In this case, as the download speed/receive throughput should be higher and should be completed without any latency, choose Associated Power Save with Low Latency with a listen interval of 1 second. Here, Fast PSP brings high Rx throughputs.
3. Consider a use case with predictable and deterministic application traffic. The application expects the SiWx917 to sleep for longer duration and perform data transmission or reception for 100-200 ms duration every 2 minutes. Beyond this 100-200 ms duration, the application shall not transmit any data. In this case, the TWT-based Power Save Mechanism can be used with a TWT Wake Interval of 2 minutes and TWT Wake Duration of 200 ms.
4. Let us consider an application with the following requirements:
   - The time at which the Rx data arrives is unpredictable. The Rx data reception is not periodic.
   - Expects an average Tx throughput of 10 Mbps with a maximum Receive Latency of 5 seconds.
   - Performs data transmissions less frequently, say 20-30 times a day.

   In this case, the Auto TWT-based Power Save Mechanism can be used which ensures longer sleep duration providing reduced current consumption as well as low latency while receiving the data that arrives at random times.
5. Consider an application that scans for Wi-Fi Networks at regular intervals to get the Wi-Fi Networks/APs statistics in the vicinity. The scan needs to be performed once a minute. In this case, the SiWx917 can be set in Standby Power Save with RAM Retention Mode when idle (for 1 minute), set to Active mode to perform scan, and can be set back to Standby Power Save with RAM Retention Mode. This process reduces SiWx917's average current consumption.
6. Consider applications that publish data to the cloud 2-3 times a day. In such cases, the device can be set to Standby Power Save Mode. Whenever data must be published, the SiWx917 can be set into Active Mode, connect to a Wi-Fi network, perform data publish, disconnect from Wi-Fi, and set back into Standby Power Save Mode.

8. SiWx917 Current Consumption Analysis

This section describes Reference examples and Setup used to measure current consumption during various stages such as during wireless scan, wireless connection, power save modes, etc.

8.1 Reference Examples

The following are the reference examples in the WiSeConnect 3 SDK for configuring Power Save Modes.

- For Connected Power Save Modes configuration, refer to the Power Save Standby Associated example.
- For Unconnected Power Save Modes configuration, refer to the Power Save Deep Sleep example.
- For TWT Power Saving Mechanism, refer to the TWT TCP Client, TWT Use Case Demo App examples.

8.2 Prerequisites

8.2.1 Hardware Used

- SiWx917 Wi-Fi 6 and Bluetooth LE Co-Processor EXP Expansion Kit
  - BRD8045A EXP Adapter Board for SiWx917 Co-Processors (hereafter referred to as Adapter Board)
  - BRD4346A SiWx917 Wi-Fi 6 and Bluetooth LE 4MB Flash Co-Processor Radio Board (here after referred to as SiWx917 NCP Radio Board)
- xG24-PK6009A EFR32xG24 +10 dBm Pro Kit
  - BRD4002A Wireless Pro Kit (WPK) Mainboard
  - BRD4186C Radio Board
- Keysight N6705C DC Power Analyzer - for measuring voltage across SiWx917 chip
- Access Point (Used only in case of Connected Power Save Modes)
- Saleae Logic Analyzer (Can be used to check the Power Save GPIOs' digital signals for understanding SiWx917's sleep/wake state switching)

8.2.2 Software Used

- Simplicity Studio IDE
- WiSeConnect SDK v3.x
- Keysight 14585A application - to be used with Keysight N6705C DC Power Analyzer to view Voltage vs Time Graphs
- Logic 2 application - to be used with Saleae Logic Analyzer to view Power Save GPIOs' digital signals

8.3 Current Measurement Methods

There are three ways to measure the current consumption of SiWx917 chip using the EXP Expansion Kit and EFR32xG24 +10 dBm Pro Kit.

- Using Advanced Energy Monitoring (AEM) System in BRD4002A Wireless Pro Kit (WPK) Mainboard
  - This method does not require external instrumentation. However, being an indirect method (refer to UG569: Adapter Board for Co-Processor Radio Boards User's Guide chap 6.2), it can only be used to estimate average currents Accuracy is good enough to evaluate currents in the range of a few µA (sleep currents).
- Using Current Sense Output pads of Adapter board and oscilloscope
  - This method is less accurate than the other methods, but it provides more bandwidth, and it might turn useful while debugging rapid changes of state, e.g., sleep-wake-sleep sequences (refer to UG569: Adapter Board for Co-Processor Radio Boards User's Guide).
- Using BRD4346A Radio Board and a current meter
  - This method can be used to connect more sophisticated external instruments If performed correctly, it will be good to reproduce datasheet figures, However, this method is prone to induce leakage currents by the host peripheral if not correct-ly handled in firmware, hence not recommended for debugging purpose.

8.3.1 AEM-Based Current Measurement

AEM is a HW feature of WPK. Energy Profiler is a graphic SW tool in the Simplicity IDE that can be used for tracing the current consumption. This method does not require any additional instruments/hardware used to measure current. In summary, it is based on two alternate current measurements read by the host board and estimates the device's consumption as difference of the two first measurements. Therefore, this method can only be used to estimate average currents.

Refer to how to use Energy Profiler. Alternatively, AEM readings are available from command line, using Simplicity Commander instead of Energy Profiler. In this way, more options are available, e.g. setting average window and more, For more details on AEM, refer to UG525: BRD4186C User Guide.

This method is based on two current measurements and estimates the device's current consumption as an indirect measurement.

Procedure:

- Switch to Energy Profiler perspective in the Simplicity Studio IDE or open the Simplicity Commander.
- Connect the EFR32xG24 WPK to the Energy Profiler in Studio.
- Configure the Adapter Board in hosted Power Mode by toggling the PWR MODE switch of Adapter Board to HOST.
- Power Cycle the EFR32xG24 host (which in turn resets the SiWx917 chip).
- If using Energy Profiler, view the Current vs Time graph and measure average current consumption over desired duration. If using Simplicity Commander, give the appropriate command. For example, for measuring the average current over a period of 30 seconds, give the following command: `commander aem measure --windowlength 30000`. This gives you total current consumption of host Board and the SiWx917. Refer this current as I(total).
- Disconnect EFR32xG24 from Energy Profiler and disconnect the EFR32xG24.
- Configure the Adapter Board in hosted Power Mode by toggling the PWR MODE switch of Adapter Board to BUF.
- Power up the EFR32xG24 and connect it to Energy Profiler, if using Energy Profiler.
- If using Energy Profiler, view the Current vs Time graph and measure average current consumption over desired duration. If using Simplicity Commander, give the respective command to obtain current reading. This gives you current consumption of host Board alone. Refer this current as I(host).
- Calculate the difference I(SiWx917) = I(total) - I(host). This gives the SiWx917's current consumption.

8.3.2 Current Measurement using Current Sense Output Pads of Adapter Board

The adapter board features two analog outputs, with different gains and bandwidths, which give a direct measure of the current absorbed by the device. This method is intended for dynamic debugging, to detect quick changes of states which might not be detected with the other methods presented in this application note and it is most suitable for oscilloscopes with more than 1MHz bandwidth to capture the dynamic. For more details, refer to the UG569: Adapter Board for Co-Processor Radio Boards User's Guide.

8.3.3 Current Measurement Using BRD4346A Radio Board

This method can be used to reproduce datasheet figures if correctly applied. The current measurement point is at the bottom side of the BRD4346A Radio board. Remove the R219 (0 ohms) resistor and connect the positive and negative terminals of the current measurement point to a current meter through appropriate connecting wires as shown below.

Figure 8.1. Current Measurement Point on Radio Board

This method presents the advantage of connecting the most accurate external instruments, but it is prone to create a potential difference between the two power domains, of respectively host and device, as it introduces a resistive element between them to sense the current. If the host IOs are not tristated, the host IOs will source some current to the device, in parallel to the current meter, altering the measurements/producing invalid results.

Choosing an advanced current meter, able to track the host voltage and dynamically replicate it at the negative sense/source terminal would be a safer option. The instrument bandwidth should allow the voltage to settle based on its internal sense resistance and the board input capacitance, about 25 µF. If not sure about these parameters, the other two methods are considered safer debugging options.

The host MCU can be as per the user's choice. In this case, the EFR32xG24 host MCU is used

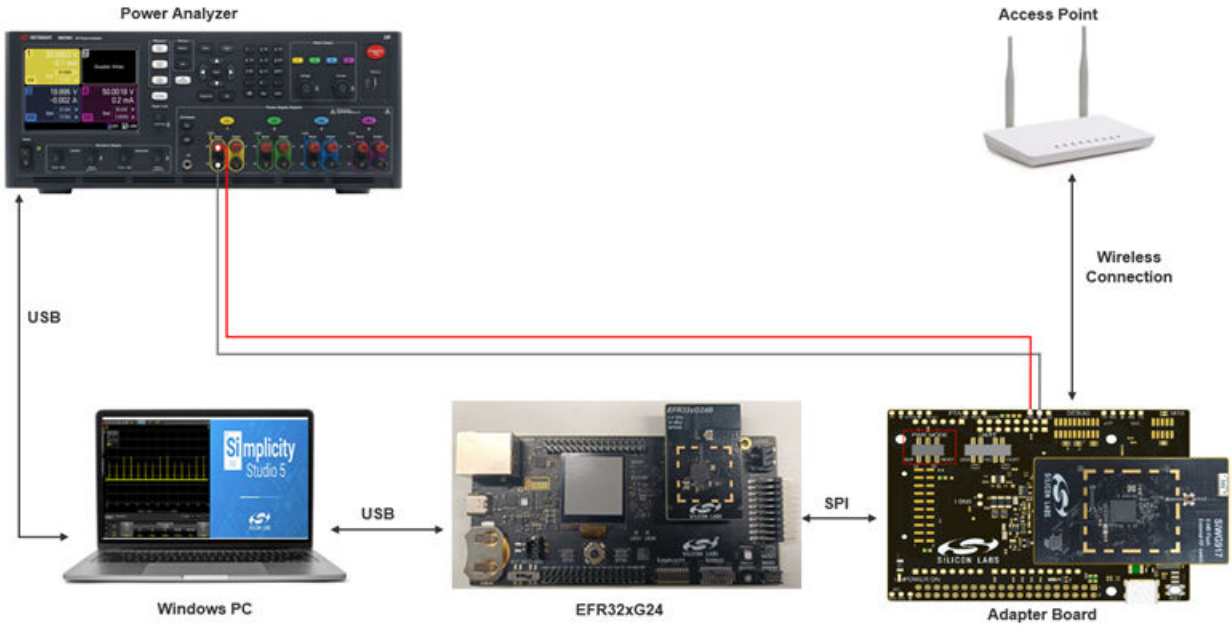

Figure 8.2. Current Measurement Using Radio Board Setup Diagram

8.4 Current Consumption Values

This section describes the current consumption of SiWx917 during different stages and states.

Note: The current consumption values provided in this section are just for reference purpose.

8.4.1 Active Mode Current Consumption

1. Device initialization

The following figure shows the Current vs Time graph and average current consumed in Keysight 14585A application obtained using Power Analyzer and current measurement point on BRD4346A Radio Board.

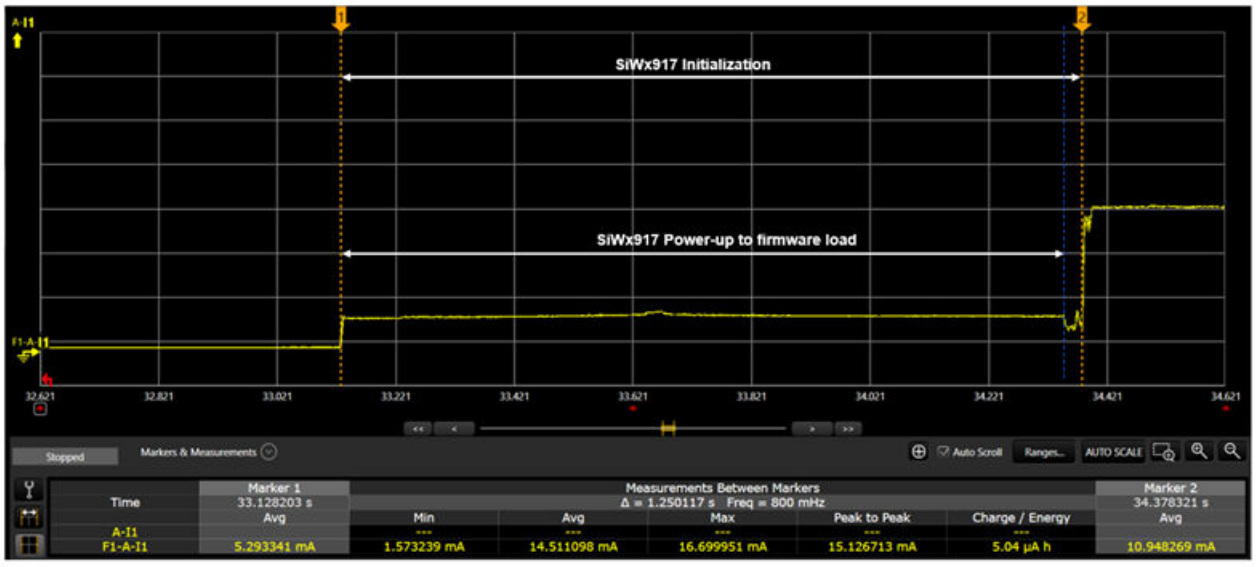

Figure 8.3. SiWx917 Device Initialization

State	Average Current Consumption	Time Taken
SiWx917's Power-up to firmware load	14.59 mA	1.21 seconds
SiWx917's Power-up to radio initialization	14.51 mA	1.25 seconds

2. Wireless Scan

By default, the applications are configured to scan on all 2.4 GHz channels (1-11) for the specified SSID. The SiWx917 sends a directed Probe Request specifying the SSID it is looking for on all the 2.4 GHz channels. The following figure shows the Current vs Time graph and average current consumed in Keysight 14585A application obtained using Power Analyzer and current measurement point on BRD4346A Radio Board.

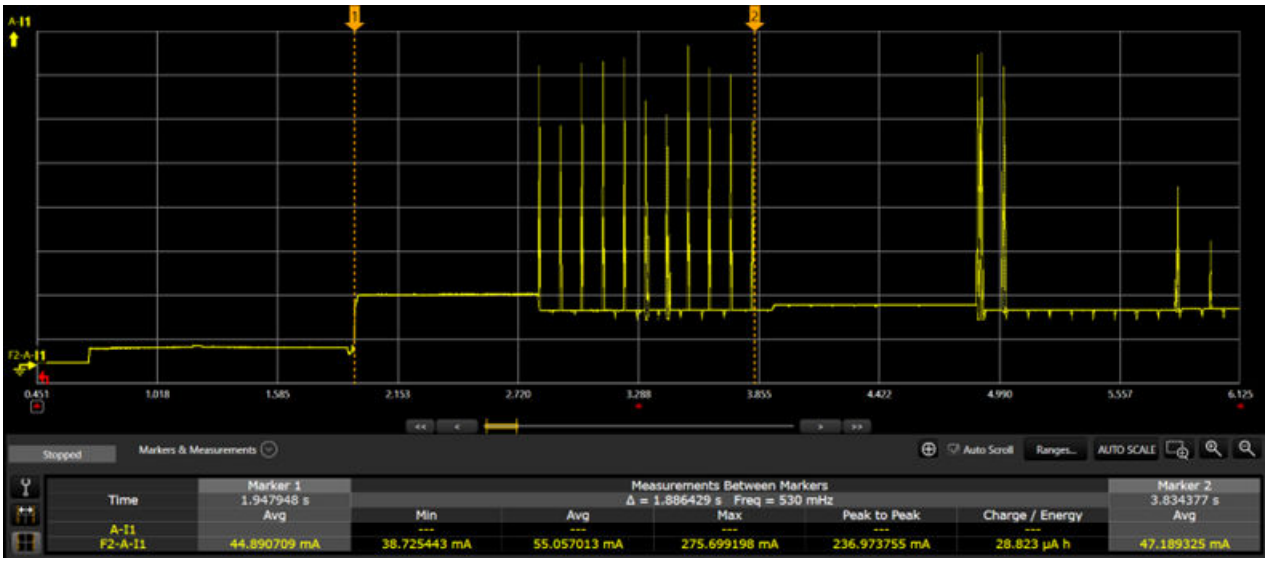

Figure 8.4. SiWx917 Current Consumption During Scan

State	Average Current Consumption	Time Taken
Wireless Scan	55 mA	~1.1 seconds

The Quick Scan feature enables SiWx917 to scan for a particular access point in a particular channel. This feature benefits you if your application connects to a known SSID on a specific channel number. This helps in reducing the time taken for scanning for APs as well as lowers the current consumption.

3. Wireless Connection

   By default, the SiWx917 sends a unicast probe request, gets a probe response, and then connects to the AP.

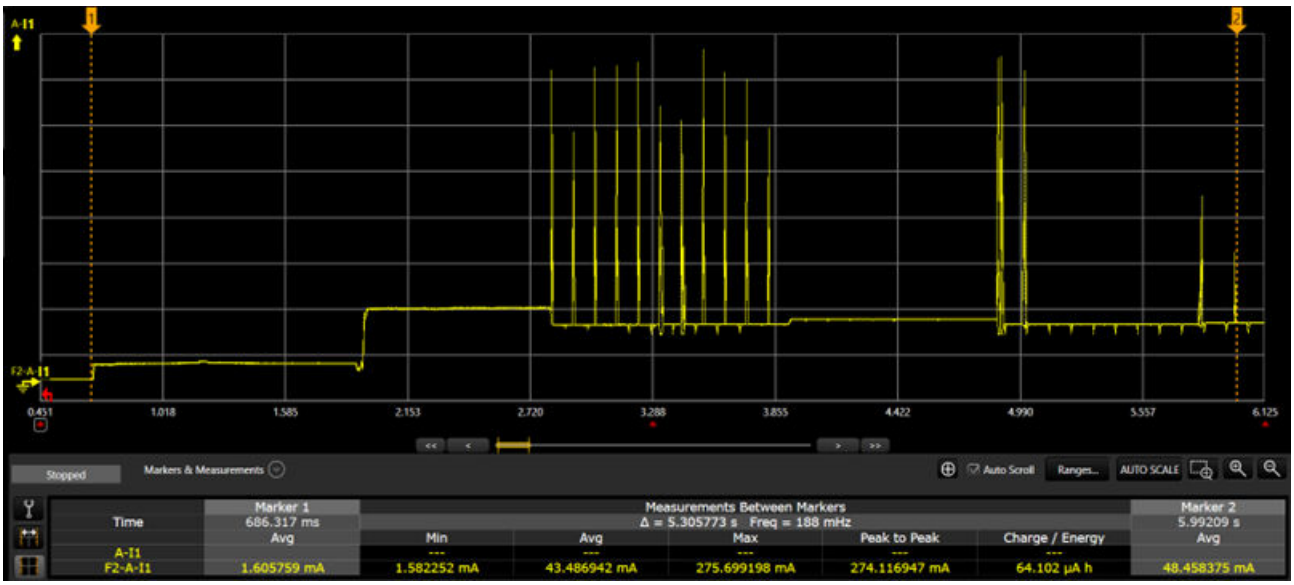

Figure 8.5. SiWx917 Current Consumption During Wi-Fi Connection

State	Average Current Consumption	Time Taken
SiWx917's Power-up to IP Configuration	43.48 mA	~5 seconds

The Quick Join feature enables SiWx917 to send authentication and association frames without unicast probe requests. To use the Quick Join feature, enable SI91X_JOIN_FEAT_QUICK_JOIN feature in the **join_feature_bitmap** and call the following API:

sl_status_t sl_si91x_set_join_configuration(sl_wifi_interface_t interface, uint8_t join_feature_bitmap)

4. Tx Active and Rx Active Currents

   The SiWx917 can be in any of the following atomic states:

   - TX_ACTIVE: Current consumption during active transmission at a given on-air data rate such as 6 Mbps, 11 Mbps, 54 Mbps, 72 Mbps, MCS7, etc., and a given output power level (e.g., 8 dBm, 18 dBm).
   - RX_ACTIVE: Current consumption during active reception at a given on-air data rate.

The above current consumption values are mentioned in **7.6.1.1 WLAN 2.4 GHz** section of the SiWx917 Data Sheet.

8.4.2 Connect Power Save Mode Current Consumption

1. **DTIM-based Power Save Current**

   Test Configuration:

   - The DTIM configured in the AP is 1.
   - The beacon interval is set to 100 ms. In terms of 802.11 Time Units (TU), the actual beacon interval is 100 ms* 1 TU = 1024 µs.

   As per the above configuration, every beacon is a DTIM beacon and SiWx917 wakes every 102.4 ms. If no listen interval configurations are enabled in the application, the SiWx917 by default wakes up based on the DTIM Period.

   The Average Current Consumption (measured over a period of two minutes) for DTIM Period 10 are mentioned in the **7.6.1.1 WLAN 2.4 GHz** section of SiWx917 Datasheet. The current consumption values in the Datasheet are for 352 KB RAM Retention configuration, which means the boot configuration, sl_wifi_device_configuration_t. .boot_config. ext_custom_feature_bit_map is set to SL_SI91X_RAM_LEVEL_NWP_BASIC_MCU_ADV. By default, the reference examples in the v3.x SDK have 672 KB RAM Retention configuration, which means sl_wifi_device_configuration_t. .boot_config. ext_custom_feature_bit_map is set to SL_SI91X_RAM_LEVEL_NWP_ALL_MCU_ZERO.

   .

   Note: The SiWx917 firmware sends WLAN Keep-alive frames every 30 seconds to maintain active connection with the connected AP.

2. **Listen Interval-based Power Save Current**

   Test Configuration:

   - The DTIM configured in the AP is 1.
   - Listen interval is set to 1000 ms.

   As per the above configuration, the SiWx917 wakes every 1000ms (every 10th DTIM beacon), which is nearest integral multiple of DTIM Period i.e., it wakes up for every 10th beacon and listens to the AP DTIM beacon frame.

   The Average Current Consumption for a Listen Interval = 1000 ms is same as DTIM period = 10.

   Examples:

   a. AP DTIM is set to 3.
   b. AP Beacon interval is 100 ms.
   c. SiWx917's Listen interval is set to 1000 ms.

   As per the above configuration, then the SiWx917 wakes every 900 ms (every 3rd DTIM beacon), which is the nearest integral multiple of DTIM Period closest to the listen interval and reads to the DTIM beacon. The more the DTIM Period, the lower the SiWx917 current consumption and vice versa.

3. **Beacon Current Profile**

When in Connected Power Save Modes, the SiWx917 listens to beacons after every DTIM period or listen- interval elapses. The following depicts the current consumption and time taken at different stages while receiving a beacon.

The following table illustrates the beacon current profile at various stages:

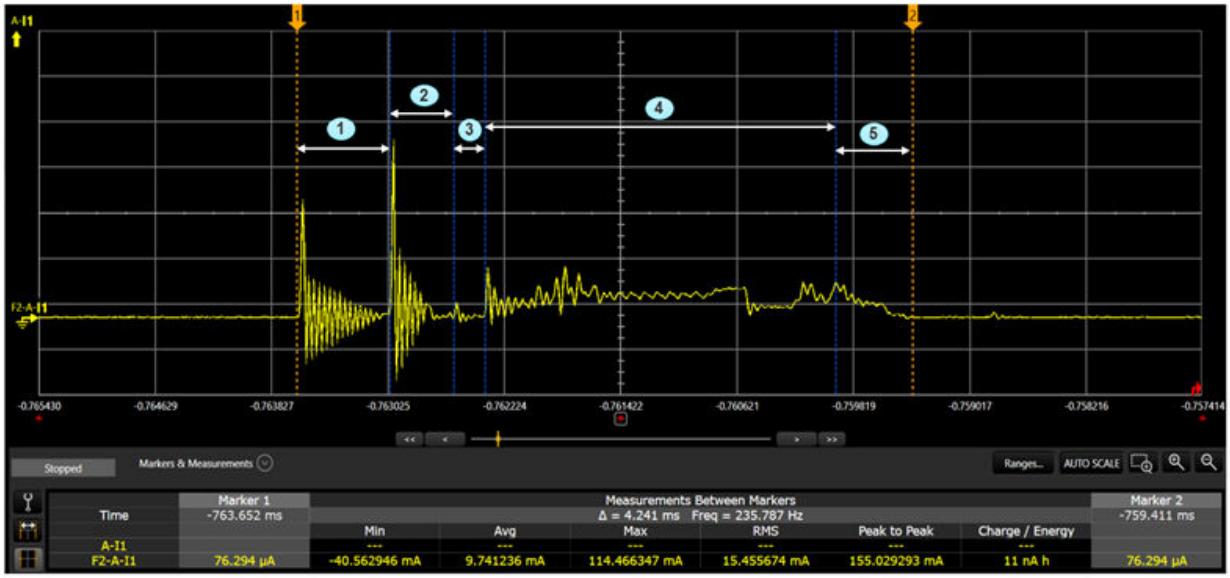

Figure 8.6. Beacon Current Profile

State	Average Current Consumption	Time Taken
Power Management Unit (PMU) Time	2.5 mA	635 µs
Crystal good Time	7 mA	443 µs
NWP and Wireless Modem Power up	1 mA	210 µs
Wireless Modem receiving a beacon in listen mode	13.2 mA	2.4 ms
NWP and Wireless Modem shut down and sleep mode	7.8 mA	457 µs

8.4.3 Typical Wake/Sleep Timings

Test Configuration:

- Connect SiWx917 to an AP.
- Set SiWx917 in Connected Power Save.
- Transmit a UDP data packet.
- For sending a UDP packet, the WiSeConnect 3 SDK driver internally sends two commands: socket create request and socket data send.
- After each command, the host gives sleep permission to SiWx917 by de-asserting the respective GPIO.
- In this case, after sending socket create request command, the host gives sleep permission to SiWx917. By the time, the SiWx917 goes to sleep, the host makes another wake request, during which time it notices that the SiWx917 is still in wake state and sends the socket data send command.

1. The following figure illustrates the amount of time taken by SiWx917 to go to sleep or wake up upon a host's request.

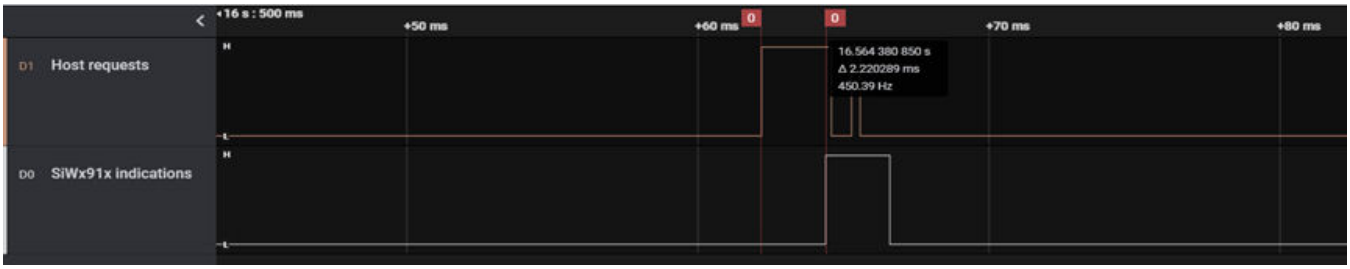

Figure 8.7. Time Taken to Wake

State	Time Taken
Time taken to Wake	2 ms
Time taken to Sleep	1 ms

2. The following figure illustrates the amount of time taken by SiWx917 to go to sleep or wake upon a host's request.

Figure 8.8. Time Taken to Sleep

9. Low Power and Interoperability Considerations

- It is recommended to set the device into Connected Power Save Mode only after IP configuration
- If a Wireless disconnection happens when the SiWx917 is in Power Save Mode, disable the power save and try to reconnect to the AP.
- To avoid interoperability issues with various APs, enable the **Enhanced Max PSP feature**.
- For applications, where throughput is not a major concern, consider disabling the higher data rates (MCS5, MCS6, and MCS7). To do this, make sure BIT(19) - SL_SI91X_FEAT_DISABLE_MCS_5_6_7_DATARATES in **config_feature_bit_map** is not enabled in the boot configuration.
- Make a smart configuration of WLAN Keep-Alive, TCP Keep-Alive, and MQTT Keep-Alive parameters as per your application to reduce the current consumption.
- In power save modes, if the DNS requests fail with few APs, the SL_SI91X_FEAT_AGGREGATION in **feature_bit_map** is to be enabled in boot configuration. If this does not help, it is recommended to disable the power save and then make a DNS request API call and configure the device back into power save mode.
- If broadcast/multicast data is not important for your application, further power consumption can be reduced using the broadcast filter command.

10. Sections to be Added in the Upcoming Versions

- Details of “Off” state.
- Updated current consumption values in power save modes with various APs.
- Updated current consumption measurement techniques.
- Current Consumption vs Listen Interval.
- Channel Utilization vs Current Consumption.
- Filter broadcast API usage and details.
- The Current consumption when using the Quick Scan feature and Quick Join feature.
- Wake interval aligned with beacon vs wake interval aligned with DTIM beacon.
- More details on TWT.

11. Revision History

Revision 1.0

February, 2024

- Initial Revision

Smart. Connected. Energy-Friendly.

IoT Portfolio
www.silabs.com/products

Quality
www.silabs.com/quality

Support & Community
www.silabs.com/community

SILICON LABS

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

www.silabs.com