UG190: EZR32HG 868 MHz 16 dBm Wireless Starter Kit User's Guide

A Wireless Starter Kit with the BRD4546A Radio Board is an excellent starting point to get familiar with the EZR32™ Happy Gecko Wireless Microcontroller, and it provides all necessary tools for developing a Silicon Labs wireless application.

BRD4546A is a plug-in board for the Wireless Starter Kit Mainboard. It is a complete reference design for the EZR32HG Wireless MCU, with matching network for 16 dBm output power, and an SMA connector for the 868 MHz band. The radio board also features a USB Micro-B connector with device mode support.

The Wireless Starter Kit Mainboard contains an on-board J-Link debugger with a Packet Trace Interface and a Virtual COM port, enabling application development and debugging the attached radio board as well as external hardware. The Mainboard also contains sensors and peripherals for easy demonstration of some of the EZR32's many capabilities.

This document describes how to use the BRD4546A Radio Board together with a Wireless Starter Kit Mainboard.

BRD4546A RADIO BOARD FEATURES

- EZR32 Happy Gecko Wireless MCU with 64 kB Flash, and 8 kB RAM. (EZR32HG320F64R68G)
- 16 dBm output power
- SMA connector for 868 MHz RF
- USB 2.0 Full Speed (12 Mbps) Device Mode

WIRELESS STK MAINBOARD FEATURES

- Advanced Energy Monitor
- Packet Trace Interface
- Virtual COM Port
- SEGGER J-Link on-board debugger
- External device debugging
- Ethernet and USB connectivity
- Silicon Labs Si7021 Relative Humidity and Temperature sensor
- Low-power 128x128 pixel Memory LCD
- User LEDs / Pushbuttons
- 20-pin 2.54 mm EXP header
- Breakout pads for Wireless MCU I/O
- CR2032 coin cell battery support

SOFTWARE SUPPORT

- Simplicity Studio™
- Energy Profiler
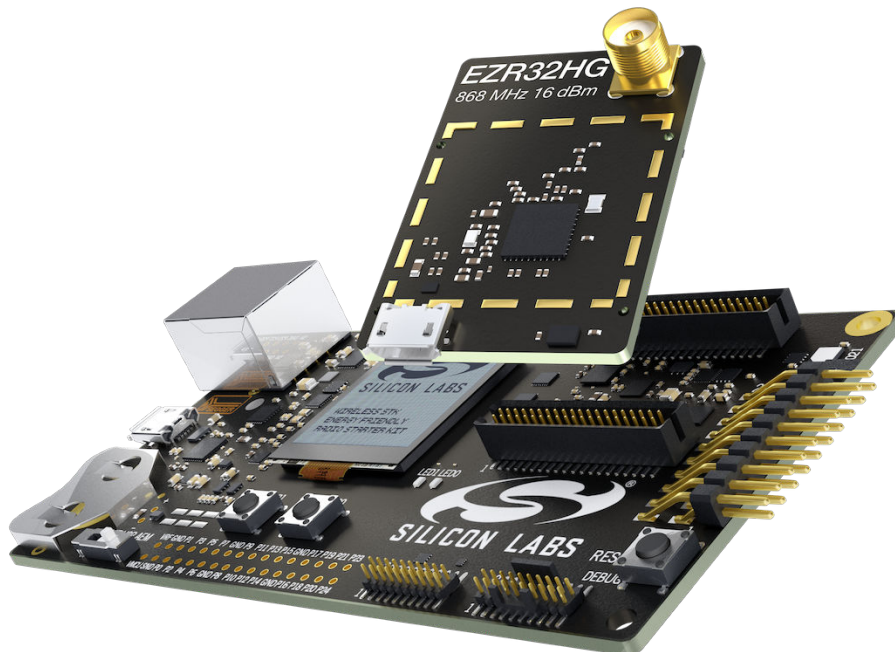
ORDERING INFORMATION

- SLWSTK6240A

Table of Contents

1. Introduction

The EZR32HG Happy Gecko Wireless MCU is featured on a radio board that plugs directly into a Wireless Starter Kit Mainboard. The mainboard features several tools for easy evaluation and development of wireless applications. An on-board J-Link debugger enables programming and debugging on the target device over USB or Ethernet. The Advanced Energy Monitor (AEM) offers real-time current and voltage monitoring. A virtual COM port interface (VCOM) provides an easy-to-use serial port connection over USB or Ethernet. The Packet Trace Interface (PTI) offers invaluable debug information about transmitted and received packets in wireless links.

All debug functionality, including AEM, VCOM, and PTI, can also be used towards external target hardware instead of the attached radio board.

To further enhance its usability, the mainboard contains sensors and peripherals that demonstrate some of the many capabilities of the EZR32HG. The mainboard also has a 20-pin EXP header which can be used for connecting EXP boards to the kit or for easy connection to I/Os on the target IC on the radio board.

1.1 Radio Boards

A Wireless STK consists of one or more mainboards and radio boards that plug into the mainboard. Different radio boards are available, each featuring different Silicon Labs devices with different operating frequency bands.

Because the mainboard is designed to work with different radio boards, the actual pin mapping from a device pin to a mainboard feature is done on the radio board. This means that each radio board has its own pin mapping to the Wireless STK features, such as buttons, LEDs, the display, the EXP header and the breakout pads. Because this pin mapping is different for every radio board, it is important to consult the correct document, which shows the kit features in context of the radio board plugged in.

This document explains how to use the Wireless STK when the EZR32HG 868 MHz 16 dBm Radio Board (BRD4546A) is combined with a Wireless STK Mainboard. The combination of these two boards is hereby referred to as a Wireless Starter Kit (Wireless STK).

1.2 Ordering Information

BRD4546A can be obtained as part of the SLWSTK6240A Wireless Starter Kit.

Table 1.1. Ordering Information

Part Number	Description	Contents
SLWSTK6240A	EZR32HG 868 MHz Wireless Starter Kit	2x BRD4001A Wireless Starter Kit Mainboard
		2x BRD4546A EZR32HG 868 MHz 16 dBm Radio Board
		2x 868 MHz dipole antenna (Linx ANT-868-CW-HWR-SMA)
		2x USB Type A to Micro-B cable
		2x USB Type A to Mini-B cable
SLWRB4546A[1]	EZR32HG 868 MHz 16 dBm Radio Board	1x BRD4546A EZR32HG 868 MHz 16 dBm Radio Board
		1x 868 MHz dipole antenna (Linx ANT-868-CW-HWR-SMA)

1   Discontinued.

1.3 Getting Started

Detailed instructions for how to get started can be found on the Silicon Labs web pages: http://www.silabs.com/dev-tools.

2. Hardware Overview

2.1 Hardware Layout

The layout of the EZR32HG 868 MHz 16 dBm Wireless STK is shown in the figure below.
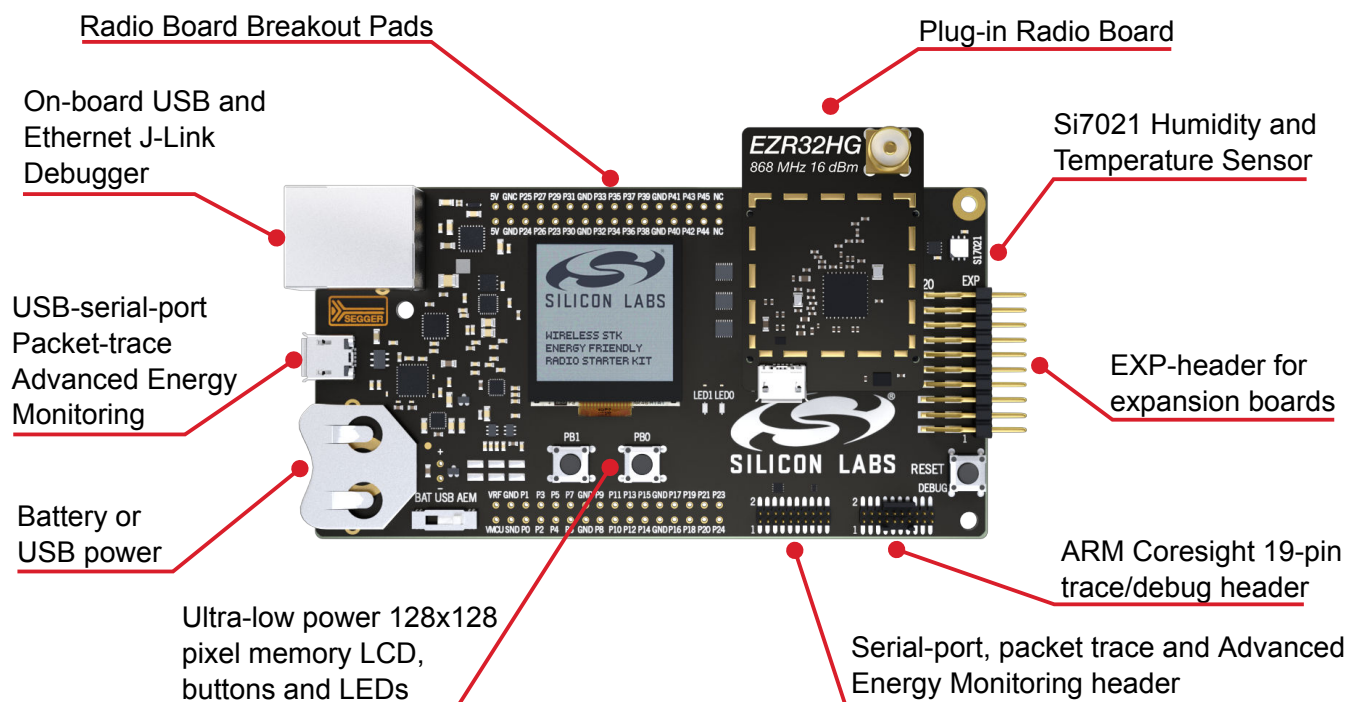
Figure 2.1. Kit Hardware Layout

2.2 Block Diagram

An overview of the EZR32HG 868 MHz 16 dBm Wireless STK is shown in the figure below.
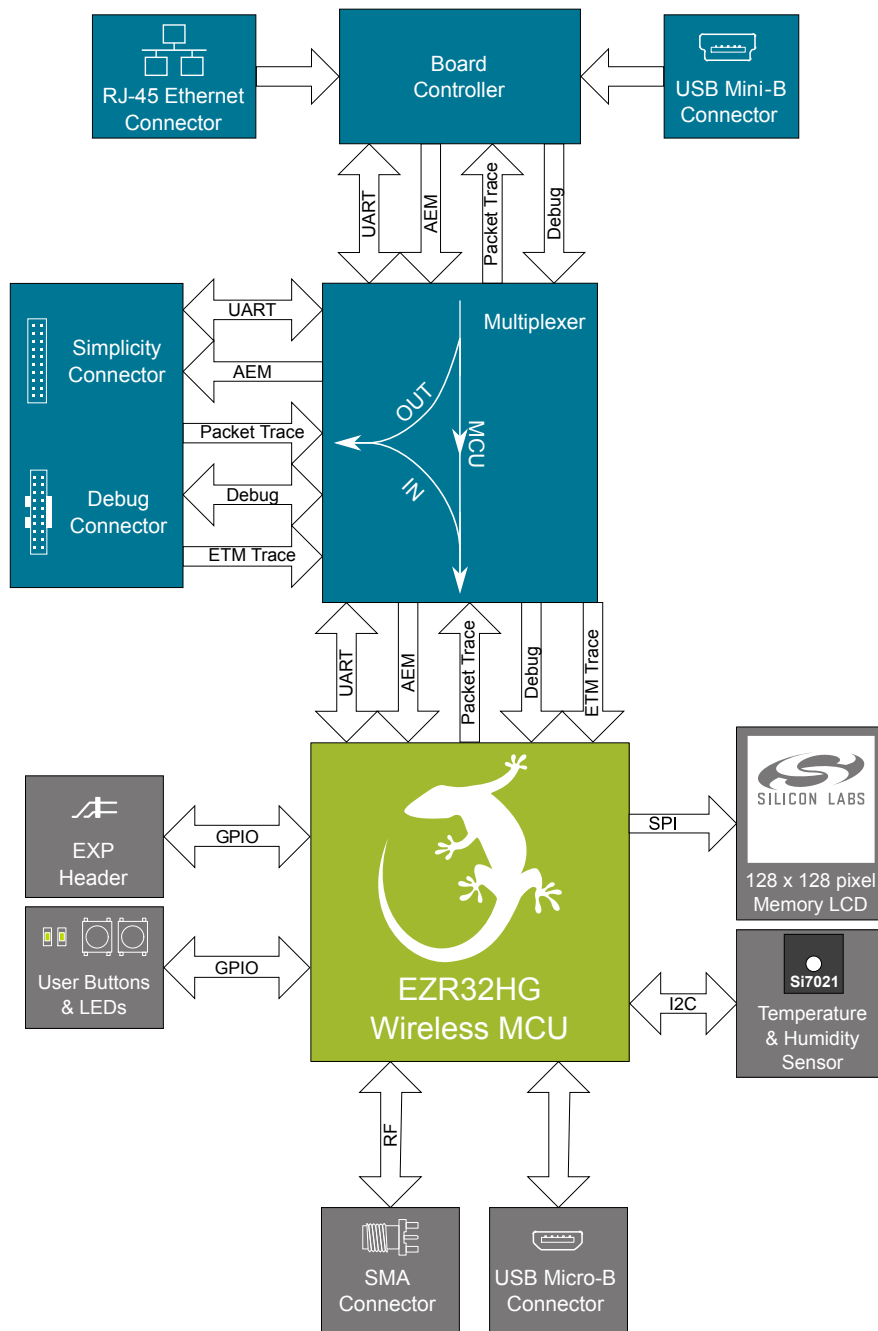
Figure 2.2.   Kit Block Diagram

3. Connectors

This chapter gives you an overview of the Wireless STK Mainboard connectivity. The placement of the connectors is shown in the figure below.
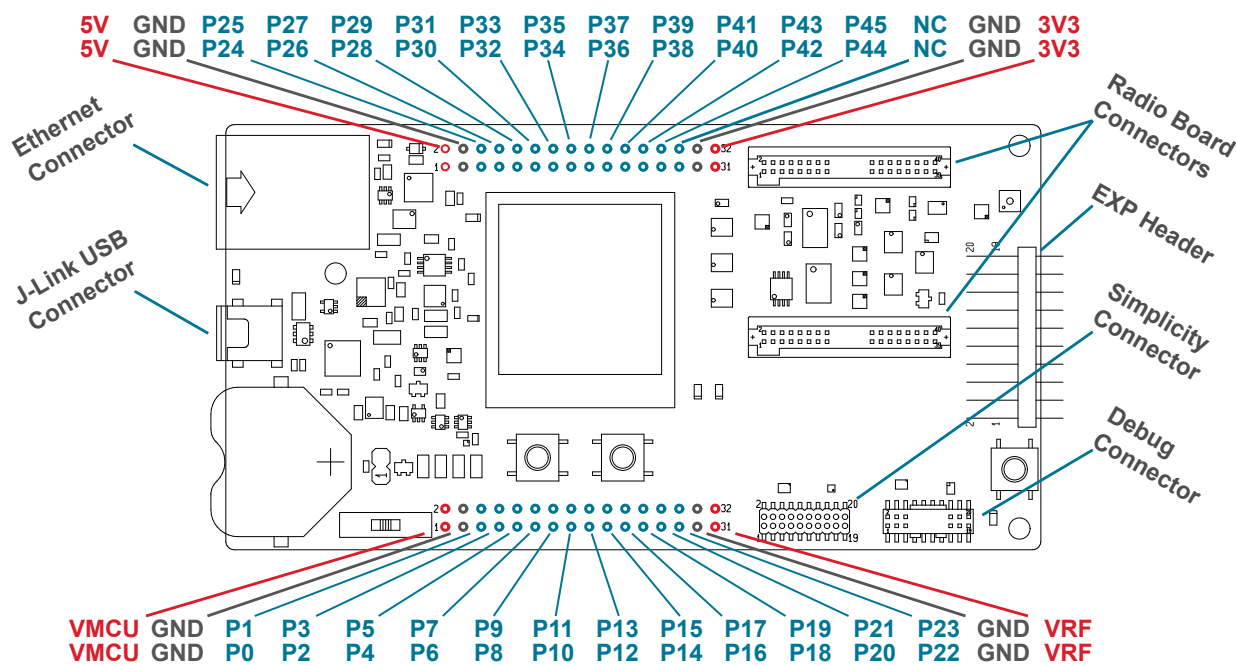
Figure 3.1. Mainboard Connector Layout

3.1 J-Link USB Connector

The J-Link USB connector is situated on the left side of the mainboard and provides access to the kit features described in Section 6. Board Controller through the USB interface. In addition to providing access to development features of the kit, this USB connector is also the main power source for the kit powering both the board controller and the AEM as described in Section 4. Power Supply and Reset.

3.2 Ethernet Connector

The Ethernet connector is situated on the left side of the mainboard and provides access to the kit features described in Section 6. Board Controller over TCP/IP. The J-Link USB connector must be connected while using this interface to provide power to the Wireless STK as power is not supplied over the Ethernet connector.

3.3 Breakout Pads

Most of the EZR32 pins are routed from the radio board to breakout pads at the top and bottom edges of the Wireless STK Mainboard. A 2.54 mm pitch pin header can be soldered on for easy access to the pins. The figure below shows you how the pins of the EZR32 map to the pin numbers printed on the breakout pads. To see the available functions on each, refer to the data sheet for EZR32HG320F64R68G.

```
                                    BOTTOM EDGE
                                 VMCU  o o  VMCU
                                  GND  o o  GND
                   LED1 / PF2 / EXP3 / P0  o o  P1 / EXP4 / PE10 / DISP_SI
                   LED0 / PF4 / EXP5 / P2  o o  P3 / EXP6 / PE11
                   BTN0 / PC9 / EXP7 / P4  o o  P5 / EXP8 / PE12 / DISP_SCLK
                  BTN1 / PC10 / EXP9 / P6  o o  P7 / EXP10 / PE13
                       PB11 / EXP11 / P8  o o  P9 / EXP12 / PD4 / VCOM_TX
                        NC / EXP13 / P10  o o  P11 / EXP14 / PD5 / VCOM_RX
              I2C_SCL / PD7 / EXP15 / P12  o o  P13 / EXP16 / PD6 / I2C_SDA
                    DISP_CS / PA0 / P14  o o  P15 / PA1 / DISP_ENABLE
                             NC / P16  o o  P17 / NC
                             NC / P18  o o  P19 / PC8 / VCOM_ENABLE
           DBG_TCK_SWCLK / PF0 / P20  o o  P21 / DBG_TMS_SWDIO / PF1
              DISP_EXTCOMIN / PF3 / P22  o o  P23 / NC
                                  GND  o o  GND
                                  VRF  o o  VRF


                                     TOP EDGE
                                   5V  o o  5V
                                  GND  o o  GND
                              NC / P24  o o  P25 / NC
                              NC / P26  o o  P27 / NC
                              NC / P28  o o  P29 / NC
            PTI_DATA / RF_GPIO0 / P30  o o  P31 / RF_GPIO1 / PTI_FRAME
                    RF_GPIO2 / P32  o o  P33 / RF_GPIO3
                              NC / P34  o o  P35 / NC
                              NC / P36  o o  P37 / Tied High / SENSOR_ENABLE
                              NC / P38  o o  P39 / NC
                              NC / P40  o o  P41 / NC
                              NC / P42  o o  P43 / NC
                              NC / P44  o o  P45 / NC
                                   NC  o o  NC
                                  GND  o o  GND
                                  3V3  o o  3V3
```
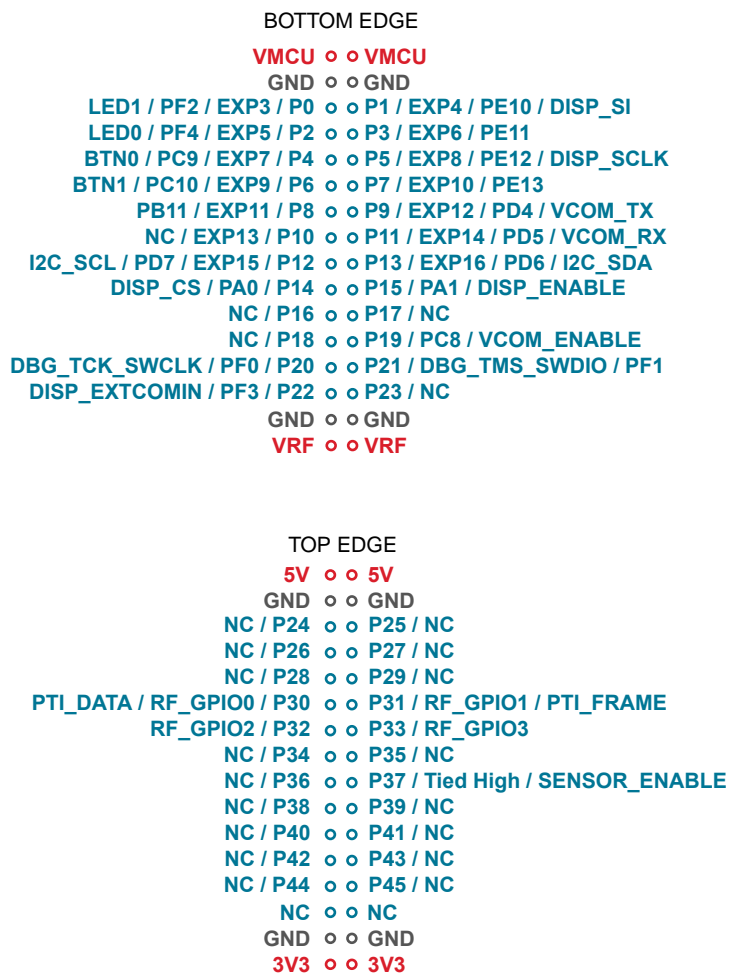
Figure 3.2. Breakout Pad Pin Mapping

3.4 EXP Header

The EXP header is an angled, 20-pin expansion header that allows connection of peripherals or plugin boards to the kit. It is located on the right-hand side of the mainboard and contains several I/O pins that can be used with most of the EZR32 Happy Gecko's features. Additionally, the VMCU, 3V3, and 5V power rails are also exposed.

The connector follows a standard which ensures that commonly used peripherals, such as a SPI, a UART, and an I2C bus, are available on fixed locations in the connector. The rest of the pins are used for general purpose IO. This allows the definition of expansion boards (EXP boards) that can plug into several different Silicon Labs Starter Kits.

The figure below shows the pin assignment of the EXP header. Because of limitations in the number of available GPIO pins, some of the EXP header pins are shared with kit features.
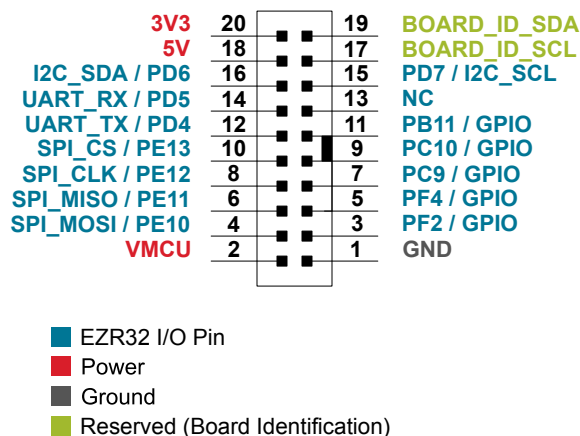
3V3	20		19	BOARD_ID_SDA
5V	18		17	BOARD_ID_SCL
I2C_SDA / PD6	16		15	PD7 / I2C_SCL
UART_RX / PD5	14		13	NC
UART_TX / PD4	12		11	PB11 / GPIO
SPI_CS / PE13	10		9	PC10 / GPIO
SPI_CLK / PE12	8		7	PC9 / GPIO
SPI_MISO / PE11	6		5	PF4 / GPIO
SPI_MOSI / PE10	4		3	PF2 / GPIO
VMCU	2		1	GND

■ EZR32 I/O Pin
■ Power
■ Ground
■ Reserved (Board Identification)

Figure 3.3. EXP Header

3.4.1 Expansion Header Pin-out

Many pins on the EZR32 are shared between the Expansion Header and other functions on the Wireless STK Mainboard. includes an overview of the mainboard features that share pins with the Expansion Header.

Table 3.1. Expansion Header Pinout

Pin	Connection	EXP Header function	Shared feature	Peripheral mapping
20	3V3	Board controller supply		
18	5V	Board USB voltage		
16	PD6	I2C_SDA	SENSOR_I2C_SDA	I2C0_SDA #1
14	PD5	UART_RX	VCOM_RX_MISO	LEUART0_RX #0
12	PD4	UART_TX	VCOM_TX_MOSI	LEUART0_TX #0
10	PE13	SPI_CS		USART0_CS #0
8	PE12	SPI_SCLK	DISP_SCLK	USART0_CLK #0
6	PE11	SPI_MISO		USART0_RX #0
4	PE10	SPI_MOSI	DISP_MOSI	USART0_TX #0
2	VMCU	EZR32 voltage domain, included in AEM measurements.		
19	BOARD_ID_SDA	Connected to Board Controller for identification of add-on boards.		
17	BOARD_ID_SCL	Connected to Board Controller for identification of add-on boards.		
15	PD7	I2C_SCL	SENSOR_I2C_SCL	I2C0_SCL #1
13	Not connected			
11	PB11	GPIO		
9	PC10	GPIO	BUTTON1	
7	PC9	GPIO	BUTTON0	
5	PF4	GPIO	LED0	
3	PF2	GPIO	LED0	
1	GND	Ground		

3.5 Debug Connector

The debug connector serves multiple purposes based on the "debug mode" setting which can be configured in Simplicity Studio. When the debug mode is set to "Debug IN", the debug connector can be used to connect an external debugger to the EZR32 on the radio board. When set to "Debug OUT", this connector allows the kit to be used as a debugger towards an external target. When set to "Debug MCU" (default), the connector is isolated from both the on-board debugger and the radio board target device.

Because this connector is electronically switched between the different operating modes, it can only be used when the board controller is powered (i.e., J-Link USB cable connected). If debug access to the target device is required when the board controller is unpowered, connect directly to the appropriate breakout pins.

The pinout of the connector follows that of the standard ARM Cortex Debug+ETM 19-pin connector. The pinout is described in detail below. Even though the connector has support for both JTAG and ETM Trace, it does not necessarily mean that the kit or the on-board target device supports these features.
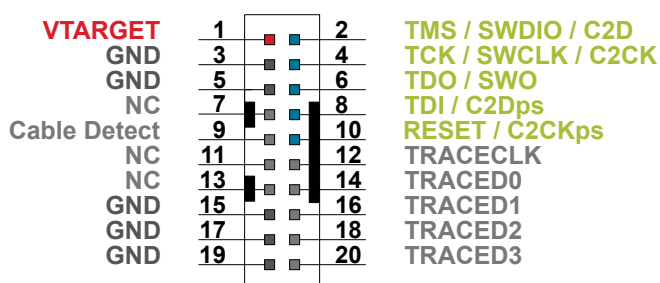
VTARGET	1		2	**TMS / SWDIO / C2D**
GND	3		4	**TCK / SWCLK / C2CK**
GND	5		6	**TDO / SWO**
NC	7		8	**TDI / C2Dps**
Cable Detect	9		10	**RESET / C2CKps**
NC	11		12	TRACECLK
NC	13		14	TRACED0
GND	15		16	TRACED1
GND	17		18	TRACED2
GND	19		20	TRACED3

Figure 3.4. Debug Connector

Note: The pinout matches the pinout of an ARM Cortex Debug+ETM connector, but these are not fully compatible because pin 7 is physically removed from the Cortex Debug+ETM connector. Some cables have a small plug that prevents them from being used when this pin is present. If this is the case, remove the plug or use a standard 2x10 1.27 mm straight cable instead.

Table 3.2. Debug Connector Pin Descriptions

Pin Number(s)	Function	Description
1	VTARGET	Target reference voltage. Used for shifting logical signal levels between target and debugger.
2	TMS / SDWIO / C2D	JTAG test mode select, Serial Wire data, or C2 data
4	TCK / SWCLK / C2CK	JTAG test clock, Serial Wire clock, or C2 clock
6	TDO/SWO	JTAG test data out or Serial Wire Output
8	TDI / C2Dps	JTAG test data in or C2D "pin sharing" function
10	RESET / C2CKps	Target device reset or C2CK "pin sharing" function
12	TRACECLK	Not connected
14	TRACED0	Not connected
16	TRACED1	Not connected
18	TRACED2	Not connected
20	TRACED3	Not connected
9	Cable detect	Connect to ground
7, 11, 13	NC	Not connected
3, 5, 15, 17, 19	GND	Ground

Note: Although the on-board debugger and the Debug Connector supports JTAG, the EZR32HG-series of devices do not support JTAG.

3.6 Simplicity Connector

The Simplicity Connector enables the advanced debugging features, such as the AEM, the virtual COM port, and the Packet Trace Interface, to be used towards an external target. The pinout is illustrated in the figure below.
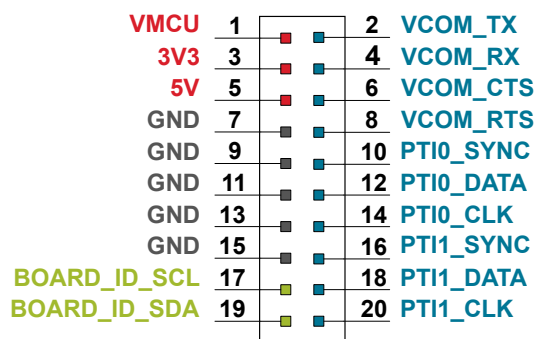
VMCU	1	2	VCOM_TX
3V3	3	4	VCOM_RX
5V	5	6	VCOM_CTS
GND	7	8	VCOM_RTS
GND	9	10	PTI0_SYNC
GND	11	12	PTI0_DATA
GND	13	14	PTI0_CLK
GND	15	16	PTI1_SYNC
BOARD_ID_SCL	17	18	PTI1_DATA
BOARD_ID_SDA	19	20	PTI1_CLK

Figure 3.5. Simplicity Connector

Note: Current drawn from the VMCU voltage pin is included in the AEM measurements, while the 3V3 and 5V voltage pins are not. When monitoring the current consumption of an external target with the AEM, unplug the radio board from the mainboard to avoid adding the radio board's current consumption to the measurements.

Table 3.3. Simplicity Connector Pin Descriptions

Pin Number(s)	Function	Description
1	VMCU	3.3 V power rail, monitored by the AEM
3	3V3	3.3 V power rail
5	5V	5 V power rail
2	VCOM_TX	Virtual COM Tx
4	VCOM_RX	Virtual COM Rx
6	VCOM_CTS	Virtual COM CTS
8	VCOM_RTS	Virtual COM RTS
10	PTI0_SYNC	Packet Trace 0 Sync
12	PTI0_DATA	Packet Trace 0 Data
14	PTI0_CLK	Packet Trace 0 Clock
16	PTI1_SYNC	Packet Trace 1 Sync
18	PTI1_DATA	Packet Trace 1 Data
20	PTI1_CLK	Packet Trace 1 Clock
17	BOARD_ID_SCL	Board ID SCL
19	BOARD_ID_SDA	Board ID SDA
7, 9, 11, 13, 15	GND	Ground

3.7 Debug Adapter

The BRD8010A STK/WSTK Debug Adapter is an adapter board which plugs directly into the debug connector and the Simplicity Connector on the mainboard. It combines selected functionality from the two connectors to a smaller footprint 10-pin connector, which is more suitable for space-constrained designs.

For versatility, the debug adapter features three different 10-pin debug connectors:
• Silicon Labs Mini Simplicity Connector
• ARM Cortex 10-pin Debug Connector
• Silicon Labs ISA3 Packet Trace

The ARM Cortex 10-pin Debug Connector follows the standard Cortex pinout defined by ARM and allows the Starter Kit to be used to debug hardware designs that use this connector.

The ISA3 connector follows the same pinout as the Packet Trace connector found on the Silicon Labs Ember Debug Adapter (ISA3). This allows the Starter Kit to be used to debug hardware designs that use this connector.

The Mini Simplicity Connector is designed to offer advanced debug features from the Starter Kit on a 10-pin connector:
• Serial Wire Debug (SWD) with SWO
• Packet Trace Interface (PTI)
• Virtual COM port (VCOM)
• AEM monitored voltage rail

Note: Packet Trace is only available on Wireless STK Mainboards and Wireless Pro Kit Mainboards. MCU Starter Kits do not support Packet Trace.

```
VAEM   1        2  GND
RESET  3        4  VCOM_RX
VCOM_TX 5       6  SWO
SWDIO  7        8  SWCLK
PTI_FRAME 9    10  PTI_DATA
```
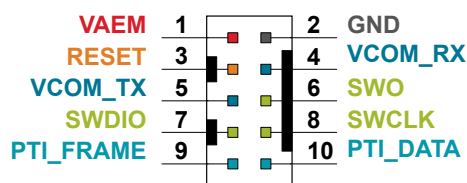
Figure 3.6. Mini Simplicity Connector

Table 3.4. Mini Simplicity Connector Pin Descriptions

Pin Number	Function	Description
1	VAEM	Target voltage on the debugged application. Supplied and monitored by the AEM when power selection switch is in the "AEM" position.
2	GND	Ground
3	RST	Reset
4	VCOM_RX	Virtual COM Rx
5	VCOM_TX	Virtual COM Tx
6	SWO	Serial Wire Output
7	SWDIO	Serial Wire Data
8	SWCLK	Serial Wire Clock
9	PTI_FRAME	Packet Trace Frame Signal
10	PTI_DATA	Packet Trace Data Signal

4. Power Supply and Reset

4.1 Radio Board Power Selection

The EZR32 on a Wireless STK can be powered by one of these sources:

• The debug USB cable
• A 3 V coin cell battery
• A USB regulator on the radio board (for devices with USB support only)

The power source for the radio board is selected with the slide switch in the lower left corner of the Wireless STK Mainboard. The figure below shows how the different power sources can be selected with the slide switch.
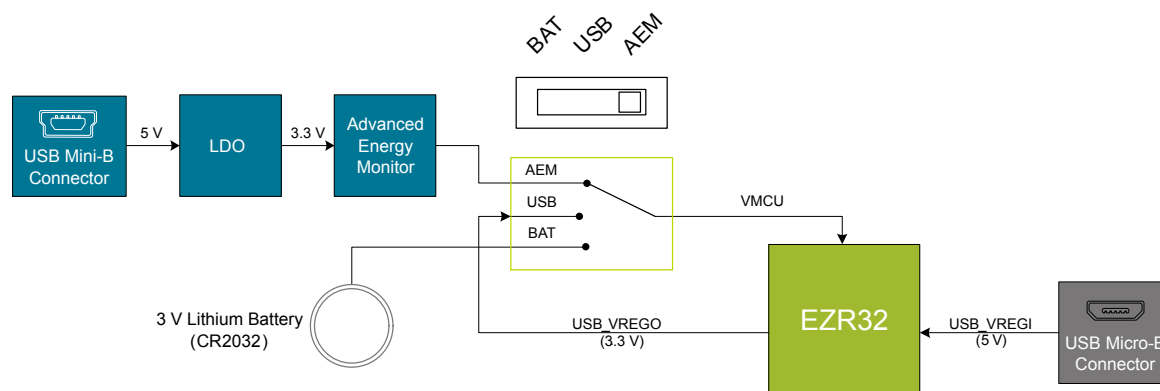
Figure 4.1. Power Switch

With the switch in the **AEM** position, a low noise 3.3 V LDO on the mainboard is used to power the radio board. This LDO is again powered from the debug USB cable. The AEM is now also connected in series, allowing accurate high speed current measurements and energy debugging/profiling.

With the switch in the **USB** position, the integrated linear regulator in the EZR32 is used to power the radio board. This allows a USB device application where the Wireless MCU operates as a bus powered device.

Finally, with the switch in the **BAT** position, a 20 mm coin cell battery in the CR2032 socket can be used to power the device. With the switch in this position, no current measurements are active. This is the switch position that should be used when the radio board is powered with an external power source.

Note: The current sourcing capabilities of a coin cell battery might be too low to supply certain wireless applications.

Note: The AEM can only measure the current consumption of the EZR32 when the power selection switch is in the **AEM** position.

4.2 Board Controller Power

The board controller is responsible for important features, such as the debugger and the AEM, and is powered exclusively through the USB port in the top left corner of the board. This part of the kit resides on a separate power domain, so a different power source can be selected for the target device while retaining debugging functionality. This power domain is also isolated to prevent current leakage from the target power domain when power to the Board Controller is removed.

The board controller power domain is not influenced by the position of the power switch.

The kit has been carefully designed to keep the board controller and the target power domains isolated from each other as one of them powers down. This ensures that the target EZR32 device will continue to operate in the **USB** and **BAT** modes.

4.3 EZR32 Reset

The EZR32 Wireless MCU can be reset by a few different sources:

• A user pressing the RESET button
• The on-board debugger pulling the #RESET pin low
• An external debugger pulling the #RESET pin low

In addition to the reset sources mentioned above, a reset to the EZR32 will also be issued during board controller boot-up. This means that removing power to the board controller (unplugging the J-Link USB cable) will not generate a reset but plugging the cable back in will as the board controller boots up.

5. Peripherals

The Wireless STK has a set of peripherals that showcase some of the EZR32 features.

Note that most EZR32 I/Os routed to peripherals are also routed to the breakout pads or the EXP header, which must be taken into consideration when using these I/Os.

5.1 USB Micro-B Connector

The BRD4546A radio board is equipped with a USB Micro-B connector that is connected directly to the EZR32 Happy Gecko. shows how the USB lines are connected to the EZR32.
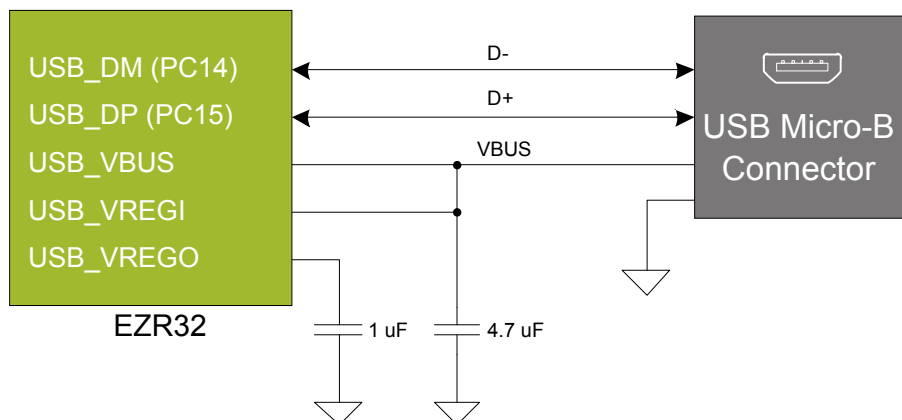
Figure 5.1. Radio Board USB Connector

Note: The Radio Board supports operation in USB Device mode only, even if EZR32HG devices also support USB Host mode.

5.2 Push Buttons and LEDs

The kit has two user push buttons marked BTN0 and BTN1. They are connected directly to the EZR32 and are debounced by RC filters with a time constant of 1 ms. The buttons are connected to pins PC9 and PC10.

The kit also features two yellow LEDs marked LED0 and LED1 that are controlled by GPIO pins on the EZR32. The LEDs are connected to pins PF4 and PF2 in an active-high configuration.
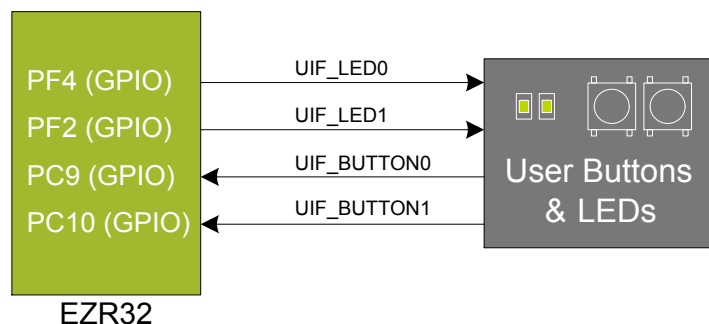
Figure 5.2. Buttons and LEDs

5.3 Memory LCD-TFT Display

A 1.28-inch SHARP Memory LCD-TFT is available on the kit to enable interactive applications to be developed. The display has a high resolution of 128 x 128 pixels and consumes very little power. It is a reflective monochrome display, so each pixel can only be light or dark, and no backlight is needed in normal daylight conditions. Data sent to the display is stored in the pixels on the glass, which means no continuous refreshing is required to maintain a static image.

The display interface consists of a SPI-compatible serial interface and some extra control signals. Pixels are not individually addressable, instead data is sent to the display one line (128 bits) at a time.

The Memory LCD-TFT display is shared with the kit's board controller, allowing the board controller application to display useful information when the user application is not using the display. The user application always controls ownership of the display with the DISP_ENABLE signal:

- DISP_ENABLE = LOW: The board controller has control of the display
- DISP_ENABLE = HIGH: The user application (EZR32) has control of the display

Power to the display is sourced from the target application power domain when the EZR32 controls the display and from the board controller's power domain when the DISP_ENABLE line is low. Data is clocked in on DISP_SI when DISP_CS is high, and the clock is sent on DISP_SCLK. The maximum supported clock speed is 1.1 MHz.

DISP_EXTCOMIN is the "COM Inversion" line. It must be pulsed periodically to prevent static build-up in the display itself. Refer to the LS013B7DH03 documentation for more information on driving the display.
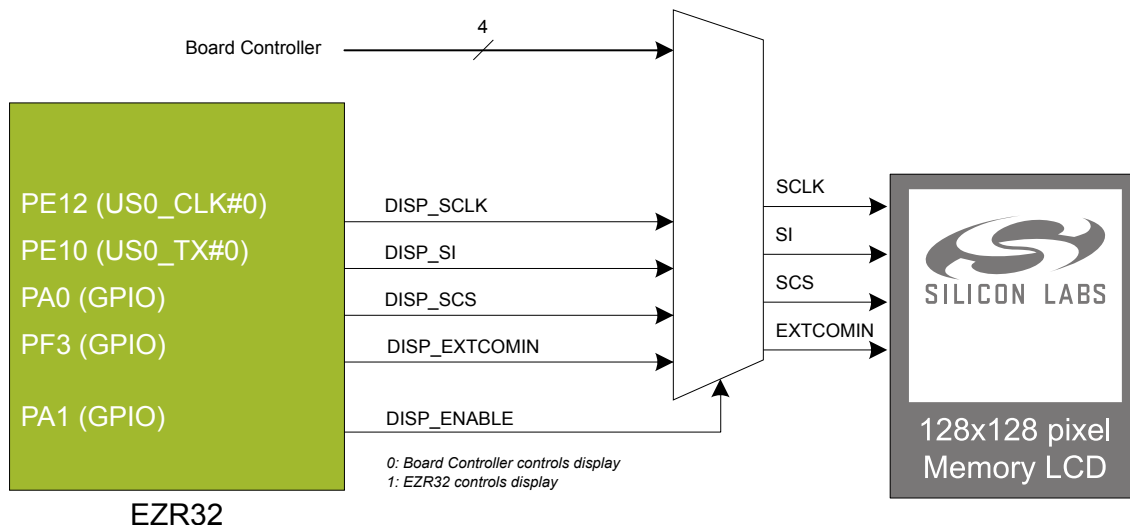

Figure 5.3. 128x128 Pixel Memory LCD

5.4 Si7021 Relative Humidity and Temperature Sensor

The Si7021 I^2C relative humidity and temperature sensor is a monolithic CMOS IC integrating humidity and temperature sensor elements, an analog-to-digital converter, signal processing, calibration data, and an I^2C Interface. The patented use of industry-standard, low-K polymeric dielectrics for sensing humidity enables the construction of low-power, monolithic CMOS Sensor ICs with low drift and hysteresis, and excellent long term stability.

The humidity and temperature sensors are factory-calibrated and the calibration data is stored in the on-chip non-volatile memory. This ensures that the sensors are fully interchangeable with no recalibration or software changes required.

The Si7021 is available in a 3x3 mm DFN package and is reflow solderable. It can be used as a hardware and software-compatible drop-in upgrade for existing RH/temperature sensors in 3x3 mm DFN-6 packages, featuring precision sensing over a wider range and lower power consumption. The optional factory-installed cover offers a low profile, convenient means of protecting the sensor during assembly (e.g., reflow soldering) and throughout the life of the product, excluding liquids (hydrophobic/oleophobic) and particulates.

The Si7021 offers an accurate, low-power, factory-calibrated digital solution ideal for measuring humidity, dew point, and temperature in applications ranging from HVAC/R and asset tracking to industrial and consumer platforms.

The I^2C bus used for the Si7021 is shared with the EXP header. The temperature sensor is normally isolated from the I^2C line. To use the sensor, SENSOR_ENABLE (tied high) must be set high. When enabled, the sensor's current consumption is included in the AEM measurements.
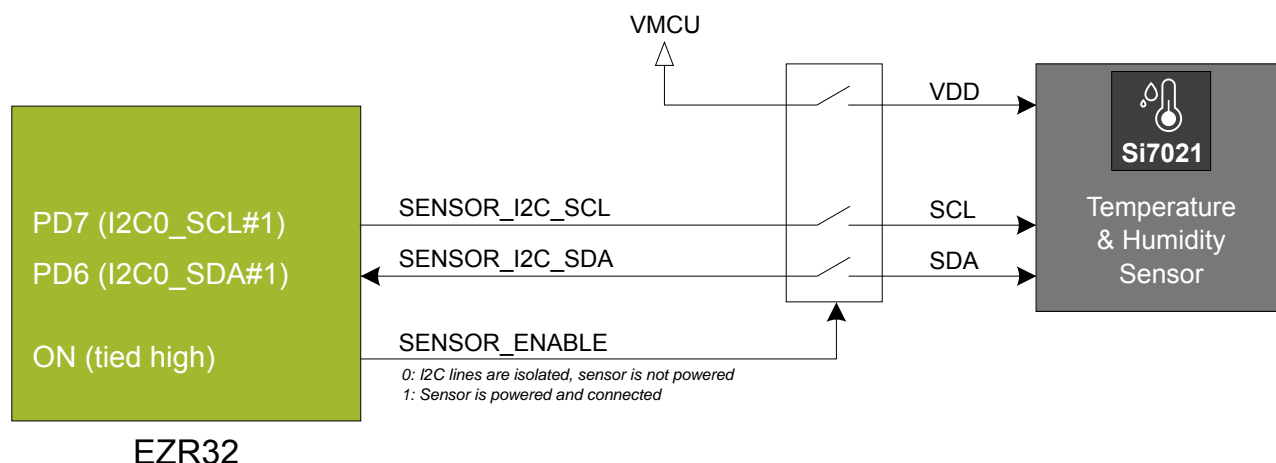

Figure 5.4. Si7021 Relative Humidity and Temperature Sensor

Refer to the Silicon Labs web pages for more information: http://www.silabs.com/humidity-sensors.

5.5 Virtual COM Port

An asynchronous serial connection to the board controller is provided for application data transfer between a host PC and the target EZR32. This eliminates the need for an external serial port adapter.
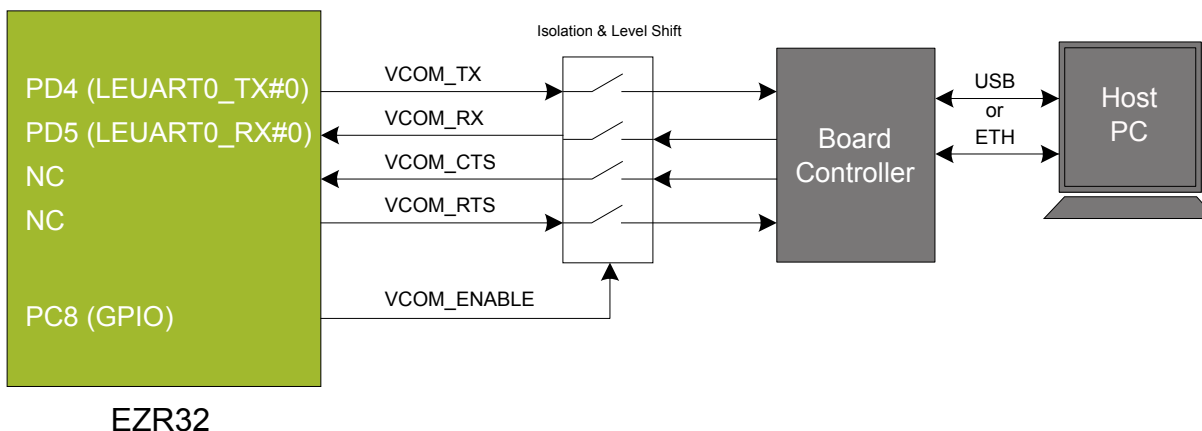

Figure 5.5. Virtual COM Port Interface

The virtual COM port consists of a physical UART between the target device and the board controller and a logical function in the board controller that makes the serial port available to the host PC over USB or Ethernet. The UART interface consists of four pins and an enable signal.

Table 5.1. Virtual COM Port Interface Pins

Signal	Description
VCOM_TX	Transmit data from the EZR32 to the board controller
VCOM_RX	Receive data from the board controller to the EZR32
VCOM_CTS	Clear to Send hardware flow control input, asserted by the board controller when it is ready to receive more data
VCOM_RTS	Request to Send hardware flow control output, asserted by the EZR32 when it is ready to receive more data
VCOM_ENABLE	Enables the VCOM interface, allowing data to pass through to the board controller

The parameters of the serial port, such as baud rate or flow control, can be configured using the admin console. The default settings depend on which radio board is used with the Wireless STK Mainboard.

Note: The VCOM port is only available when the board controller is powered, which requires the J-Link USB cable to be inserted.

5.5.1 Host Interfaces

Data can be exchanged between the board controller and the target device through the VCOM interface, which is then available to the user in two different ways:

* Virtual COM port using a standard USB-CDC driver
* TCP/IP by connecting to the Wireless STK on TCP/IP port 4901 with a Telnet client

When connecting via USB, the device should automatically show up as a COM port. The actual device name that is associated with the kit depends on the operating system and how many devices are or have been connected previously. The following are examples of what the device might show up as:

* JLink CDC UART Port (COM5) on Windows hosts
* /dev/cu.usbmodem1411 on macOS
* /dev/ttyACM0 on Linux

Data sent by the target device into the VCOM interface can be read from the COM port, and data written to the port is transmitted to the target device. Connecting to the Wireless STK on port 4901 gives access to the same data over TCP/IP. Data written into the VCOM interface by the target device can be read from the socket, and data written into the socket is transmitted to the target device.

Note: Only one of these interfaces can be used at the same time, with the TCP/IP socket taking priority. This means that if a socket is connected to port 4901, no data can be sent or received on the USB COM port.

5.5.2 Serial Configuration

By default, the VCOM serial port is configured to use 115200 8N1 (115.2 kbit/s, 8 data bits, 1 stop bit), with flow control disabled/ignored. The configuration can be changed using the admin console:

```
WSTK> serial vcom config
Usage: serial vcom config [--nostore] [handshake <rts/cts/rtscts/disable/auto>] [speed <9600,921600>]
```

Using this command, the baud rate can be configured between 9600 and 921600 bit/s, and hardware handshake can be enabled or disabled on either or both flow control pins.

5.5.3 Hardware Handshake

The VCOM peripheral supports basic RTS/CTS flow control.

VCOM_CTS (target clear to send) is a signal that is output from the board controller and input to the target device. The board controller de-asserts this pin whenever its input buffer is full and it is unable to accept more data from the target device. If hardware handshake is enabled in the target firmware, its UART peripheral will halt when data is not being consumed by the host. This implements end-to-end flow control for data moving from the target device to the host.

VCOM_CTS is connected to the RTS pin on the board controller and is enabled by setting handshake to either RTS or RTSCTS using the "serial vcom config" command.

VCOM_RTS (target request to send) is a signal that is output from the target device and input to the board controller. The board controller will halt transmission of data towards the target if the target device de-asserts this signal. This gives the target firmware a means to hold off incoming data until it can be processed. Note that de-asserting RTS will not abort the byte currently being transmitted, so the target firmware must be able to accept at least one more character after RTS is de-asserted.

VCOM_RTS is connected to the CTS pin of the board controller. It is enabled by setting handshake to either CTS or RTSCTS using the "serial vcom config" command in the admin console. If CTS flow control is disabled, the state of VCOM_RTS will be ignored and data will be transmitted to the target device anyway.

Table 5.2. Hardware Handshake Configuration

Mode	Description
disabled	RTS (VCOM_CTS) is not driven by the board controller and CTS (VCOM_RTS) is ignored.
rts	RTS (VCOM_CTS) is driven by the board controller to halt target from transmitting when input buffer is full. CTS (VCOM_RTS) is ignored.
cts	RTS (VCOM_CTS) is not driven by the board controller. Data is transmitted to the target device if CTS (VCOM_RTS) is asserted and halted when de-asserted.
rtscts	RTS (VCOM_CTS) is driven by the board controller to halt target when buffers are full. Data is transmitted to the target device if CTS (VCOM_RTS) is asserted and halted when de-asserted.

Note: Enabling CTS flow control without configuring the VCOM_RTS pin can result in no data being transmitted from the host to the target device.

6. Board Controller

6.1 Admin Console

The admin console is a command line interface to the board controller on the kit. It provides functionality for configuring the kit behavior and retrieving configuration and operational parameters.

6.1.1 Connecting

The admin console is available when the Wireless STK is connected to Ethernet using the Ethernet connector in the top left corner of the mainboard. See Section 8.1.2 Ethernet Interface for details on the Ethernet connectivity.

Connect to the admin console by opening a telnet connection to the kit's IP address, port number 4902.

When successfully connected, a `WSTK>` prompt is displayed.

6.1.2 Built-in Help

The admin console has a built-in help system which is accessed by the `help` command. The `help` command will print a list of all top level commands:

```
WSTK> help
*************** Root commands ****************
aem          AEM commands    [ calibrate, current, dump, ... ]
boardid      Commands for board ID probe.   [ list, probe ]
dbg          Debug interface status and control   [ info, mode,]
dch          Datachannel control and info commands   [ info ]
discovery    Discovery service commands.
net          Network commands.   [ dnslookup, geoprobe, ip ]
pti          Packet trace interface status and control   [ config, disable, dump, ... ]
quit         Exit from shell
sys          System commands   [ nickname, reset, scratch, ... ]
target       Target commands.   [ button, flashwrite, go, ... ]
time         Time Service commands   [ client, server ]
user         User management functions   [ login,]
```

The `help` command can be used in conjunction with any top level command to get a list of sub-commands with descriptions. For example, `pti help` will print a list of all available sub-commands of `pti`:

```
WSTK> pti help
*************** pti commands ****************
config       Configure packet trace
disable      Disable packet trace
dump         Dump PTI packets to the console as they come
enable       Enable packet trace
info         Packet trace state information
```

This means that running `pti enable` will enable packet trace.

6.1.3 Command Examples

PTI Configuration

```
pti config 0 efruart 1600000
```

Configures PTI to use the "EFRUART" mode at 1.6 Mb/s.

Serial Port Configuration

```
serial config vcom handshake enable
```

Enables hardware handshake on the VCOM UART connection.

6.2 Virtual UART

The Virtual UART (VUART) interface provides a high-performance application data interface that does not require additional I/O pins apart from the debug interface.

The Wireless STK makes the VUART interface available on TCP/IP port 4900.

6.2.1 Target-to-Host

Target-to-host communication utilizes the SWO-pin of the debug interface through the ITM debug peripheral. This approach allows a sleepy target device to enter all energy modes and still wake up intermittently to send debug information. The baud rate of the SWO data is locked to 875 kHz.

VUART utilizes ITM stimulus port 0 for general purpose printing. Silicon Labs' networking stacks utilize ITM stimulus port 8 for debug printing. The data on port 8 is encapsulated in additional framing and will also appear in the Simplicity Studio Network Analyzer.

6.2.2 Host-to-Target

Host-to-target communication utilizes SEGGER's Real Time Transfer (RTT) technology. A full explanation of how this works can be found in *J-Link/J-Trace User Guide (UM08001)*. Briefly summarized, RTT consists of a structure called the RTT Control Block, which is located in RAM. This control block points to circular buffers that the debugger can write data into. The target application can then read data out of this circular buffer.

The board controller will start searching for the RTT Control Block upon receiving data on TCP/IP port 4900. If the board controller is unable to locate the RTT Control Block, it will return an error message on the same connection. For the board controller to be able to locate the RTT Control Block, it has to be aligned on a 1024-byte boundary in RAM.

After initializing the RTT connection, the target will only enter emulated EM2 and EM3 where the power consumption remains similar to EM1. This is because RTT utilizes the debug interface, which requires use of high-frequency oscillators. Energy modes EM4S and EM4H will work as normal. When debugging energy consumption, it is therefore important to not send data on TCP/IP port 4900 as not to instantiate the RTT connection.

6.2.3 Limitations

- Because the SWO-connection can be disabled by the debugger at will, it is important for the target application to verify that SWO is enabled and configured before each transmission on the interface.
- After initializing host-to-target communication over RTT by sending data on TCP/IP port 4900, the target application will be unable to enter EM2 and EM3. This is because RTT utilizes the debug connection of the target.
- VUART might not work reliably during an active debugging session. This is because there is contention over the target's debug interface. The board controller will defer accessing the target until it is made available by the host debugger.
- VUART is designed with the assumption that only the board controller will access the RTT control block. If the target application uses RTT for other purposes, such as Segger SystemView, refrain from using VUART.

6.2.4 Troubleshooting

Problem	Solution
No data received after ending a debug session.	After certain debugger operations, the host computer manually disables SWO on the target to conserve power. This might cause SWO data to not appear if the target application initialized SWO before the debugger has disconnected. Either press the RESET button on the Wireless Starter Kit to reset the target application or make sure that the target application verifies that SWO is enabled and configured before sending any data.
No data received after flashing a new application.	
Other issues	Disconnect from TCP port 4900, press the RESET button on the kit, then reconnect to 4900. If this does not fix the issue, try to restart the kit by unplugging and replugging the USB cable.

7. Advanced Energy Monitor

7.1 Introduction

Any embedded developer seeking to make their embedded code spend as little energy as the underlying architecture supports needs tools to easily and quickly discover inefficiencies in the running application. This is what the Simplicity Energy Profiler is designed to do. In real-time, the Energy Profiler will graph and log current as a function of time while correlating this to the actual target application code running on the EZR32. There are multiple features in the profiler software that allow for easy analysis, such as markers and statistics on selected regions of the current graph or aggregate energy usage by different parts of the application.

7.2 Theory of Operation

The AEM circuitry on the board is capable of measuring current signals in the range of 0.1 µA to 95 mA, which is a dynamic range of almost 120 dB. It can do this while maintaining approximately 10 kHz of current signal bandwidth. This is accomplished through a combination of a highly capable current sense amplifier, multiple gain stages, and signal processing within the kit's board controller before the current sense signal is read by a host computer for display and/or storage.

The current sense amplifier measures the voltage drop over a small series resistor. The gain stage further amplifies this voltage with two different gain settings to obtain two current ranges. The transition between these two ranges occurs around 250 µA.

The current signal is combined with the target processor's Program Counter (PC) sampling by utilizing a feature of the ARM CoreSight debug architecture. The Instrumentation Trace Macrocell (ITM) block can be programmed to sample the MCU's PC at periodic intervals (50 kHz) and output these over SWO pin ARM devices. When these two data streams are fused and correlated with the running application's memory map, an accurate statistical profile can be built that shows the energy profile of the running application in real-time.

At kit power-up or on a power-cycle, an automatic AEM calibration is performed. This calibration compensates for any offset errors in the current sense amplifiers.
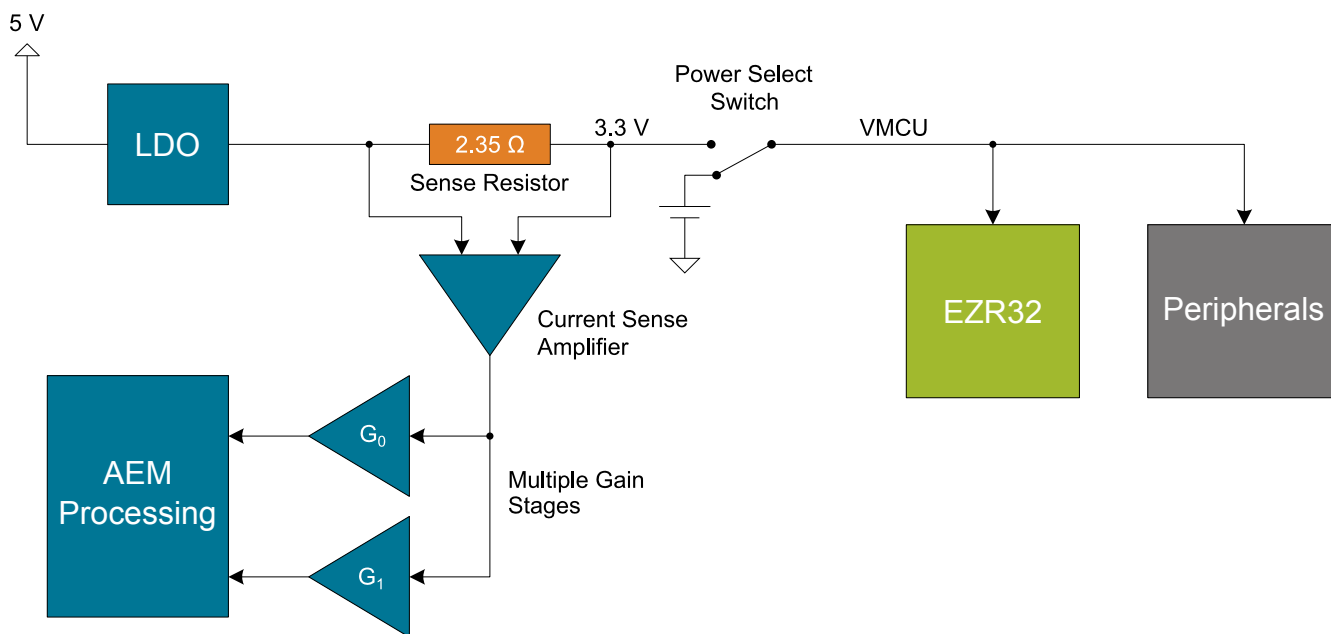
Figure 7.1. Advanced Energy Monitor

Note: The 3.3 V regulator feedback point is after the 2.35 Ω sense resistor to ensure that the VMCU voltage is kept constant when the output current changes. Maximum recommended output current is 300 mA.

7.3 AEM Accuracy and Performance

The AEM is capable of measuring currents in the range of 0.1 µA to 95 mA. For currents above 250 µA, the AEM is accurate within 0.1 mA. When measuring currents below 250 µA, the accuracy increases to 1 µA. Even though the absolute accuracy is 1 µA in the sub 250 µA range, the AEM can detect changes in the current consumption as small as 100 nA.

The AEM current sampling rate is 10 kHz.

Note: The AEM circuitry only works when the kit is powered and the power switch is in the AEM position.

7.4 Usage

The AEM data is collected by the board controller and can be displayed by the Energy Profiler, available through Simplicity Studio. By using the Energy Profiler, current consumption and voltage can be measured and linked to the actual code running on the EZR32 in realtime.

8. On-Board Debugger

The Wireless STK Mainboard contains an integrated debugger, which can be used to download code and debug the EZR32. In addition to programming a target on a plug-in radio board, the debugger can also be used to program and debug external Silicon Labs EFM32, EFM8, EZR32, and EFR32 devices connected through the debug connector.

The debugger supports three different debug interfaces for Silicon Labs devices:

 • Serial Wire Debug is supported by all EFM32, EFR32, and EZR32 devices
 • JTAG is supported by EFR32 and some EFM32 devices
 • C2 Debug is supported by EFM8 devices

For debugging to work properly, make sure the selected debug interface is supported by the target device. The debug connector on the board supports all three of these modes.

8.1 Host Interfaces

The Wireless STK supports connecting to the on-board debugger using either Ethernet or USB.

Many tools support connecting to a debugger using either USB or Ethernet. When connected over USB, the kit is identified by its J-Link serial number. When connected over Ethernet, the kit is normally identified by its IP address. Some tools also support using the serial number when connecting over Ethernet; however, this typically requires the computer and the kit to be on the same subnet for the discovery protocol (using UDP broadcast packets) to work.

8.1.1 USB Interface

The USB interface is available whenever the USB connector on the left-hand side of the mainboard is connected to a computer.

8.1.2 Ethernet Interface

The Ethernet interface is available when the mainboard Ethernet connector in the top left corner is connected to a network. Normally, the kit will receive an IP address from a local DHCP server, and the IP address is printed on the LCD display. If your network does not have a DHCP server, you need to connect to the kit via USB and set the IP address manually using Simplicity Studio, Simplicity Commander, or J-Link Configurator.

For the Ethernet connectivity to work, the kit must still be powered through the mainboard USB connector.

8.1.3 Serial Number Identification

All Silicon Labs kits have a unique J-Link serial number which identifies the kit to PC applications. This number is 9 digits and is normally on the form `44xxxxxxx`.

The J-Link serial number is normally printed at the bottom of the kit LCD display.

8.2 Debug Modes

To program external devices, use the debug connector to connect to a target board and set the debug mode to [**Out**]. The same connector can also be used to connect an external debugger to the EZR32 Wireless MCU on the kit by setting debug mode to [**In**].

Selecting the active debug mode is done in Simplicity Studio.

Debug MCU: In this mode, the on-board debugger is connected to the EZR32 on the kit.
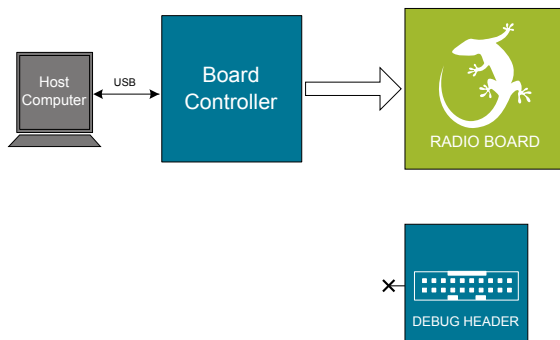
Figure 8.1. Debug MCU

Debug OUT: In this mode, the on-board debugger can be used to debug a supported Silicon Labs device mounted on a custom board.
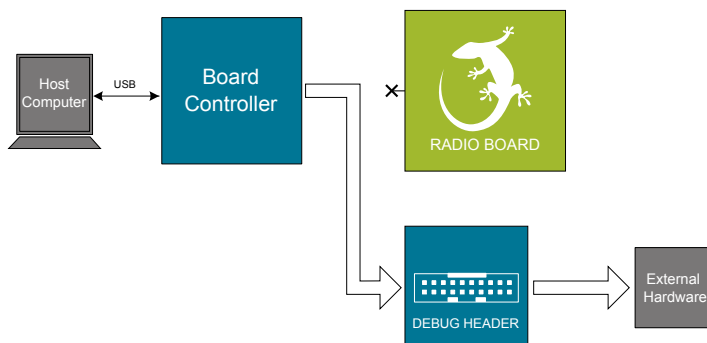
Figure 8.2. Debug OUT

Debug IN: In this mode, the on-board debugger is disconnected and an external debugger can be connected to debug the EZR32 on the kit.
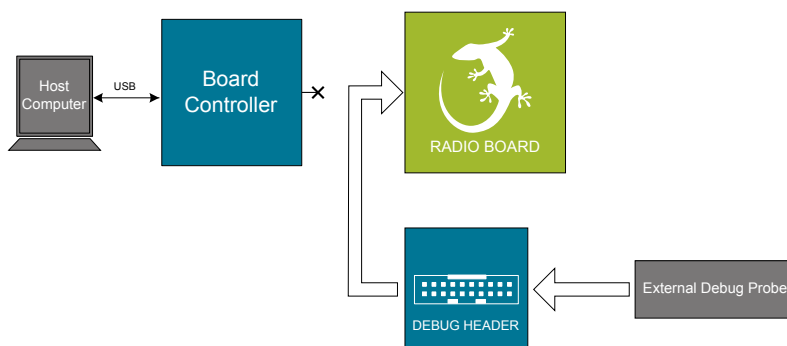
Figure 8.3. Debug IN

Note: For "Debug IN" to work, the kit board controller must be powered through the Debug USB connector.

8.3 Debugging During Battery Operation

When the EZR32 is battery-powered and the J-Link USB is still connected, the on-board debug functionality is available. If the USB power is disconnected, the Debug IN mode will stop working.

If debug access is required when the target is running off another energy source, such as a battery, and the board controller is powered down, make direct connections to the GPIOs used for debugging, which are exposed on the breakout pads.

9. Kit Configuration and Upgrades

The kit configuration dialog in Simplicity Studio allows you to change the J-Link adapter debug mode, upgrade its firmware, and change other configuration settings. To download Simplicity Studio, go to silabs.com/simplicity.

In the main window of the Simplicity Studio's Launcher perspective, the debug mode and firmware version of the selected J-Link adapter are shown. Click the [**Change**] link next to any of these settings to open the kit configuration dialog.
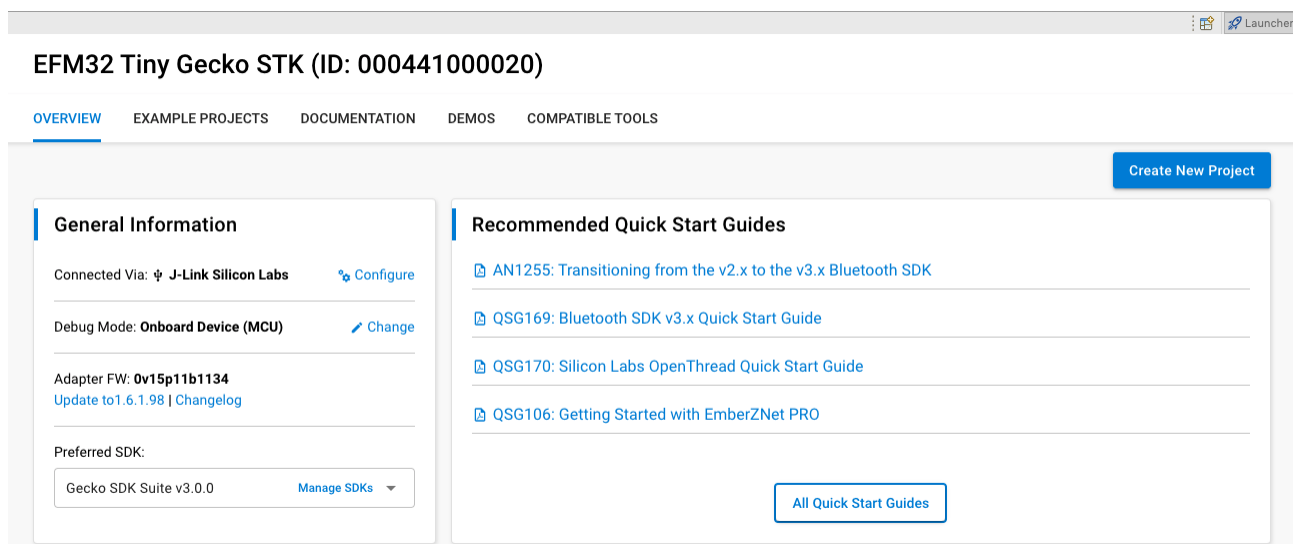
Figure 9.1. Simplicity Studio Kit Information
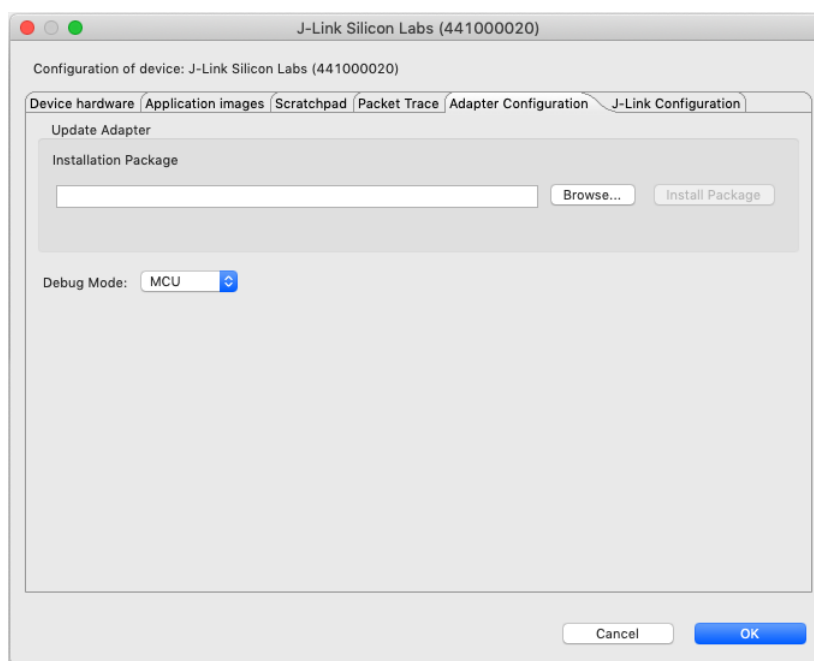
Figure 9.2. Kit Configuration Dialog

9.1 Firmware Upgrades

You can upgrade the kit firmware through Simplicity Studio. Simplicity Studio will automatically check for new updates on startup.

You can also use the kit configuration dialog for manual upgrades. Click the [**Browse**] button in the [**Update Adapter**] section to select the correct file ending in `.emz`. Then, click the [**Install Package**] button.

10. Schematics, Assembly Drawings, and BOM

Schematics, assembly drawings, and bill of materials (BOM) are available through Simplicity Studio when the kit documentation package has been installed. They are also available from the kit page on the Silicon Labs website: silabs.com.

11. Kit Revision History

The kit revision can be found printed on the kit packaging label, as outlined in the figure below. The revision history given in this section may not list every kit revision. Revisions with minor changes may be omitted.

Figure 11.1. Kit Label

11.1 SLWSTK6240A Revision History

Kit Revision	Released	Description
E00	12 January 2024	Replaced BRD4001A Wireless STK Mainboard with BRD4002A Wireless Pro Kit.
D01	15 April 2020	Updated BRD4001A Wireless STK Mainboard to Rev. A02
D00	28 September 2016	Removed coin cell battery from kit due to shipping restrictions.
C00	20 September 2016	Removed BRD8010A Debug Adapter.
B00	9 February 2016	Added BRD8010A Debug Adapter.
A00	18 December 2015	Initial kit release.

11.2 SLWRB4546A Revision History

Kit Revision	Released	Description
A00	27 April 2016	Initial release.

12. Document Revision History

Revision 1.02

January 2024

- Noted SLWRB4546A as discontinued.

Revision 1.01

8 December 2017

- Added Ordering Information section with updated kit contents for SLWSTK6240A and SLWRB4546A.
- Added SLWSTK6240A Rev B00, C00 and D00 in SLWSTK6240A Revision History section.
- Added SLWRB4546A Revision History section.

Revision 1.00

1 April 2016

- Initial document version.

Simplicity Studio

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

IoT Portfolio
www.silabs.com/IoT

SW/HW
www.silabs.com/simplicity

Quality
www.silabs.com/quality

Support & Community
www.silabs.com/community

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

www.silabs.com