UG268: EFR32BG1 2.4 GHz 8 dBm WLCSP Wireless Starter Kit User's Guide

A Silicon Labs Wireless Starter Kit for the EFR32BG1 Blue Gecko is an excellent starting point to get familiar with the device, and it provides all necessary tools for developing a Silicon Labs wireless application.

The Wireless Starter Kit Mainboard contains sensors and peripherals enabling easy demonstration of some of the EFR32™'s many capabilities. An on-board J-Link debugger allows debugging of the attached radio board as well as providing a debug connection for external hardware.

A plug-in Radio Board contains the reference design for the EFR32 itself, including the RF section and device-specific hardware.

WSTK MAINBOARD FEATURES

- Ethernet and USB connectivity
- Advanced Energy Monitor
- Virtual COM Port
- Packet Trace Interface support
- SEGGER J-Link on-board debugger
- Supports debugging the attached radio board or an external device
- Silicon Labs' Si7021 Relative Humidity and Temperature sensor
- Ultra low power 128x128 pixel Memory LCD
- User LEDs / Pushbuttons
- 20-pin 2.54 mm header for expansion boards
- Breakout pads for direct access to all radio I/O pins
- Power sources include USB, CR2032 coin cell and AA batteries.

BRD4101B RADIO BOARD FEATURES

- EFR32BG1 Blue Gecko Wireless SoC with 256 kB Flash and 32 kB RAM (EFR32BG1B232F256GJ43).
- Inverted-F PCB antenna (2.4 GHz band)
- 8 Mbit low-power serial flash for over-the-air upgrades.

SOFTWARE SUPPORT

- Simplicity Studio™
- Energy Profiler
- Network Analyzer

1. Introduction

The EFR32BG1 Blue Gecko Wireless SoC itself is featured on a Radio Board that forms a complete reference design, inluding the RF section and other components.

The EFR32 Radio Board plugs directly into a Wireless Starter Kit Mainboard. The WSTK 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 WSTK Mainboard contains sensors and peripherals demonstrating some of the EFR32BG1's many capabilities.

1.1 Radio Boards

A Wireless Starter Kit 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.

Since the mainboard is designed to work with all 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 Starter Kit 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 that the correct document be consulted which shows the kit features *in context* of the radio board plugged in.

This document explains how to use the Wireless Starter Kit when the EFR32BG1 2.4 GHz 8 dBm WLCSP Radio Board (BRD4101B) 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

BRD4101B can be obtained as a separate radio board, SLWRB4101B. The Wireless Starter Kit mainboard can be obtained as part of another kit.

Table 1.1. Ordering Information

Part Number	Description	Contents	Notes
SLWRB4101B	EFR32BG1 2.4 GHz 8 dBm WLCSP Radio Board	1x BRD4101B EFR32BG1 2.4 GHz 8 dBm WLCSP Radio Board	

1.3 Getting Started

Detailed instructions for how to get started can be found on the Silicon Labs web pages:

http://www.silabs.com/start-efr32bg

2. Hardware Overview

2.1 Hardware Layout

The layout of the EFR32BG1 2.4 GHz 8 dBm WLCSP Wireless Starter Kit is shown in the figure below.

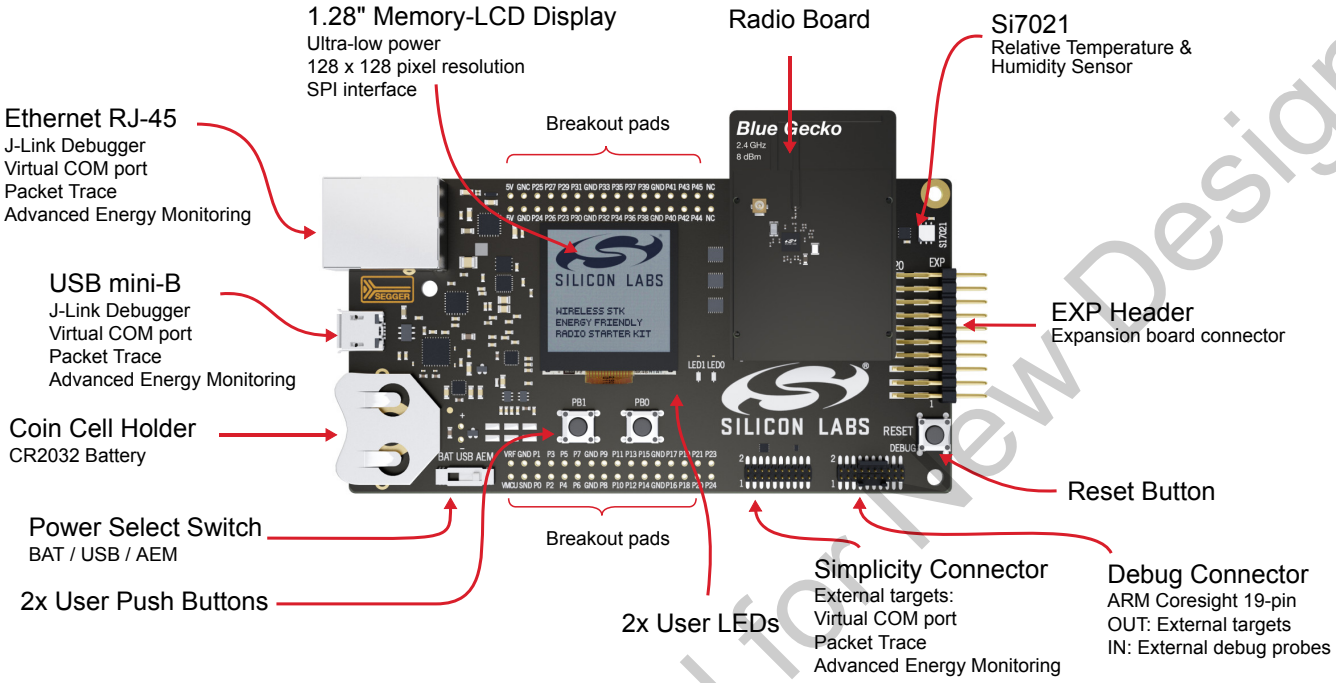

Figure 2.1. Kit Hardware Layout

2.2 Block Diagram

An overview of the EFR32BG1 2.4 GHz 8 dBm WLCSP Wireless Starter Kit is shown in the figure below.

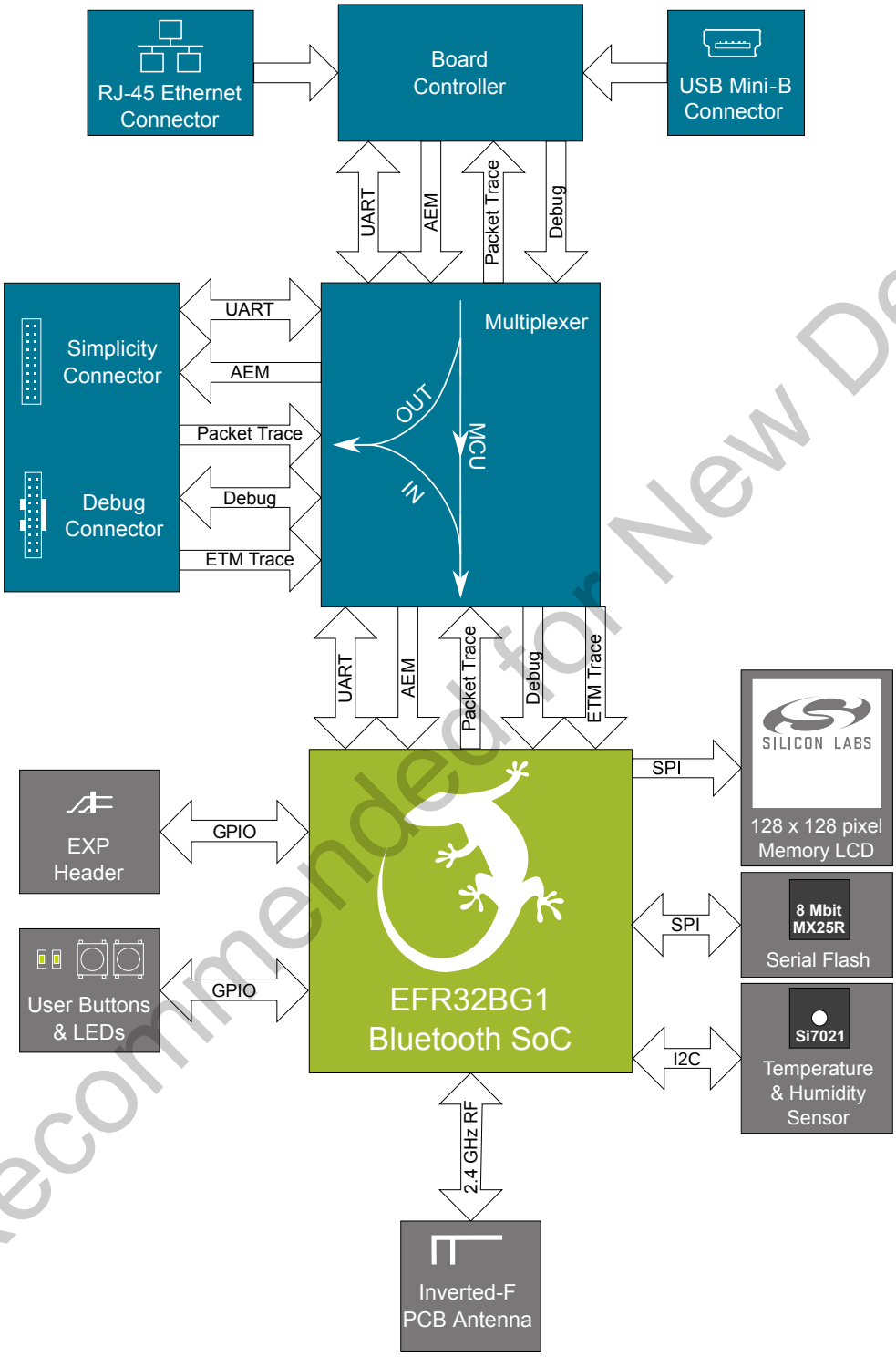

Figure 2.2. Kit Block Diagram

3. Connectors

This chapter gives you an overview of the Wireless STK Mainboard connectivity. The placement of the connectors can be seen in the figure below.

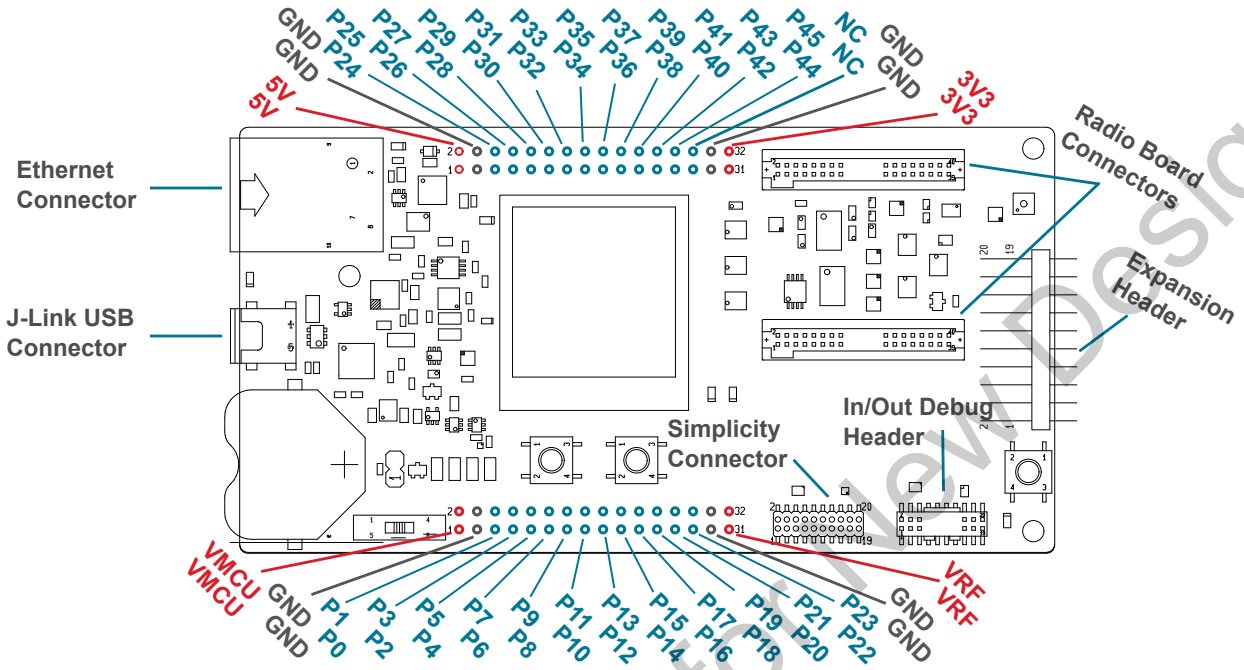

Figure 3.1. Mainboard Connector Layout

3.1 J-Link USB Connector

The J-Link USB connector is situated on the left side of the Wireless Starter Kit mainboard. Most of the kit's development features are supported through this USB interface when connected to a host computer, including:

- Debugging and programming of the target device using the on-board J-Link debugger
- Communication with the target device over the virtual COM port using USB-CDC
- Accurate current profiling using the Advanced Energy Monitor

In addition to providing access to development features of the kit, this USB connector is also the main power source for the kit. USB 5V from this connector powers the board controller and the Advanced Energy Monitor. It is recommended that the USB host be able to supply at least 500 mA to this connector, although the actual current required will vary depending on the application.

3.2 Ethernet Connector

The Ethernet connector provides access to all of the Wireless Starter Kit's development features over TCP/IP. The Ethernet interface provides some additional development features to the user. Supported features include:

- Debugging and programming of the target device using the on-board J-Link debugger
- Communication with the target device over the virtual COM port using TCP/IP socket 4901
- "VUART" communication with the target device over the debug SWD/SWO interface using TCP/IP socket 4900
- Accurate current profiling using the Advanced Energy Monitor
- Packet Trace interface supports real-time radio packet and network analysis
- The "Admin Console", a telnet console that gives access to advanced configuration options, using TCP/IP socket 4902

Please note that the Wireless Starter Kit cannot be powered using the Ethernet connector, so in order to use this interface, the USB connector must be used to provide power to the board.

3.3 Breakout Pads

Most of the EFR32's 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 EFR32 maps to the pin numbers printed on the breakout pads. To see the available functions on each, please refer to the EFR32BG1B232F256GJ43 Data Sheet.

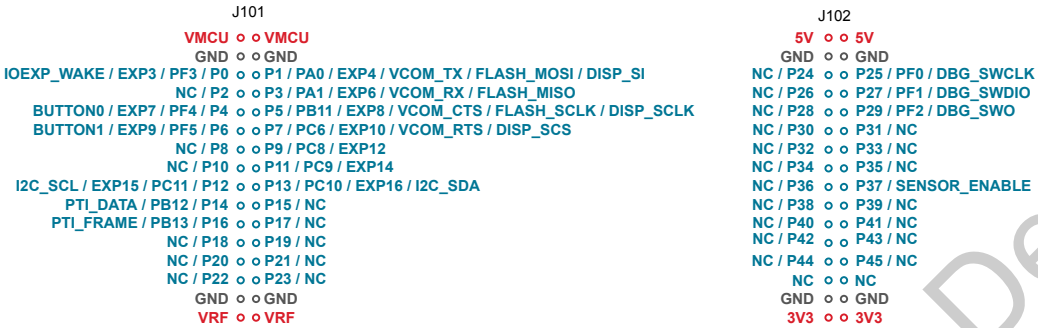

Figure 3.2. Radio Board Pin Mapping on Breakout Pads

3.4 Expansion Header

On the right hand side of the Wireless STK Mainboard an angled 20-pin expansion header is provided to allow connection of peripherals or plugin boards. The connector contains a number of I/O pins that can be used with most of the EFR32 Blue Gecko's features. Additionally, the VMCU, 3V3 and 5V power rails are also exported.

The connector follows a standard which ensures that commonly used peripherals such as an 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 that can plug into a number of different Silicon Labs Starter Kits.

The figure below shows the pin assignment of the expansion header. Because of limitations in the number of available GPIO pins, some of the expansion header pins are shared with kit features.

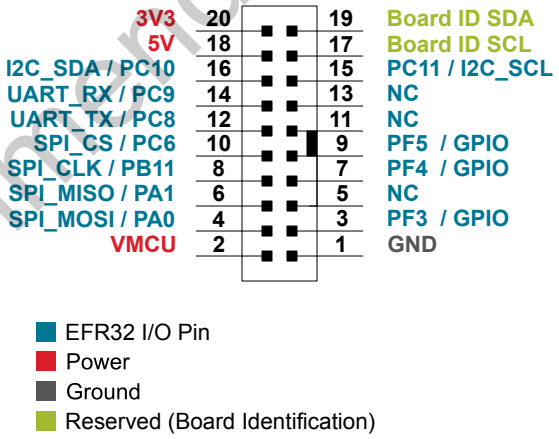

Figure 3.3. Expansion Header

3.4.1 Expansion Header Pin-out

The pin-routing on the EFR32 is very flexible, so most peripherals can be routed to any pin. However, many pins are shared between the Expansion Header and other functions on the Wireless STK Mainboard. Table 3.1 Expansion Header Pinout on page 6 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	PC10	I2C_SDA	SENSOR_I2C_SDA	I2C0_SDA #15
14	PC9	UART_RX	-	USART1_RX #13
12	PC8	UART_TX	-	USART1_TX #13
10	PC6	SPI_CS	VCOM_RTS, DISP_SCS	USART0_CS #8
8	PB11	SPI_SCLK	VCOM_CTS, DISP_SCLK, FLASH_SCLK	USART0_CLK #4
6	PA1	SPI_MISO	VCOM_RX, FLASH_MISO	USART0_RX #0
4	PA0	SPI_MOSI	VCOM_TX, DISP_SI, FLASH_MOSI	USART0_TX #0
2	VMCU	EFR32 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	PC11	I2C_SCL	SENSOR_I2C_SCL	I2C0_SCL #15
13	NC	-	-	
11	NC	-	-	
9	PF5	GPIO	BUTTON1	
7	PF4	GPIO	BUTTON0	
5	NC	-	-	
3	PF3	GPIO	IOEXP_WAKE	
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 EFR32 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 this.

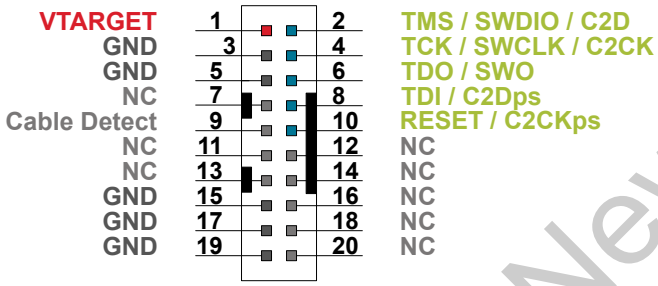

Figure 3.4. Debug Connector

Note: The pinout matches the pinout of an ARM Cortex Debug+ETM connector, but these are not fully compatible as pin 7 is physically removed from the Cortex Debug+ETM connector. Some cables have a small plug that prevent 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	
14	TRACED0	
16	TRACED1	
18	TRACED2	
20	TRACED3	
9	Cable detect	Connect to ground
11, 13	NC	Not connected
3, 5, 15, 17, 19	GND	Ground

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.

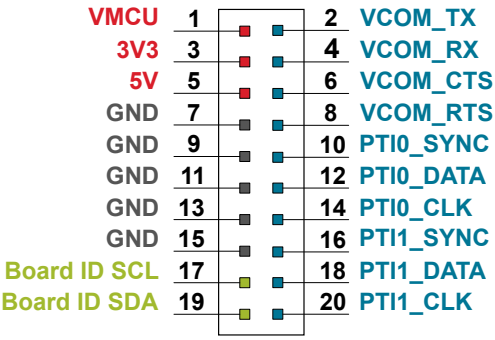

Figure 3.5. Simplicity Connector

Note: Current drawn from the VMCU voltage pin is included in the AEM measurements, while the 3V3 and 5V voltage pins are not. To monitor the current consumption of an external target with the AEM, unplug the Radio Board from the Wireless STK Mainboard to avoid that the Radio Board current consumption is added 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	EXT_ID_SCL	Board ID SCL
19	EXT_ID_SDA	Board ID SDA
7, 9, 11, 13, 15	GND	Ground

3.7 Debug Adapter

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

For versatility, the Debug Adapter feature 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 pin-out 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 pin-out 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. MCU Starter Kits do not support Packet Trace.

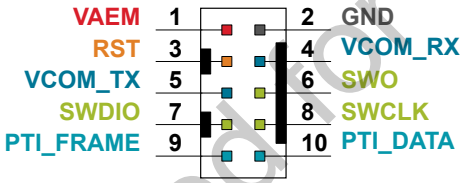

Figure 3.6. Mini Simplicity Connector

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	
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 EFR32 on a Wireless Starter Kit can be powered by one of these sources:

- the debug USB cable;
- a 3 V coin cell battery; or
- 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. Figure 4.1 Power Switch on page 10 shows how the different power sources can be selected with the slide switch.

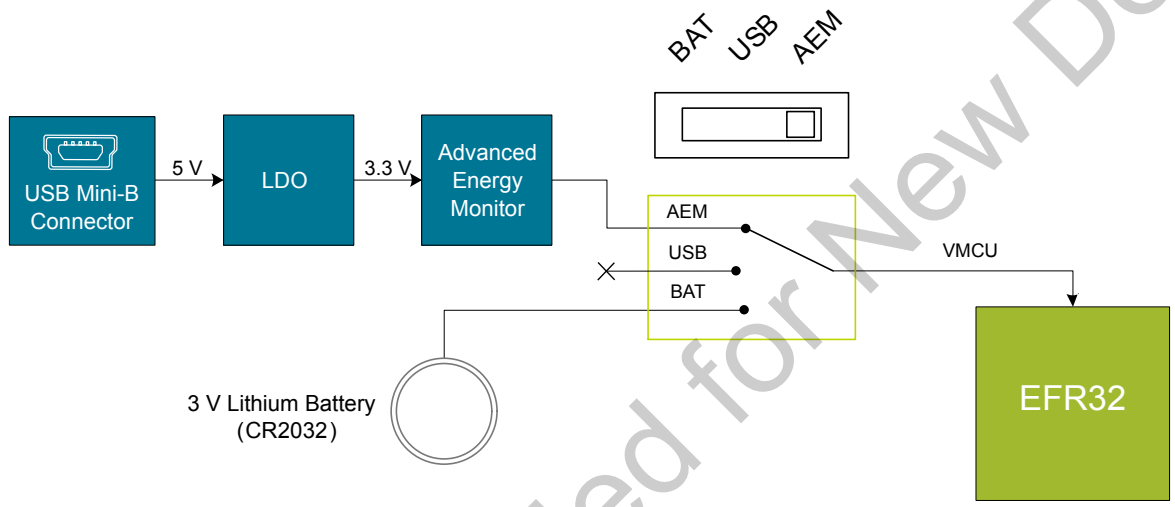

Figure 4.1. Power Switch

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

With the switch in the **USB** position, radio boards with USB-support can be powered by a regulator on the radio board itself. BRD4101B does not contain an USB regulator, and setting the switch in the **USB** postition will cause the EFR32 to be unpowered.

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 recommended switch position when powering the radio board 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 Advanced Energy Monitor can only measure the current consumption of the EFR32 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 Advanced Energy Monitor, 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 EFR32 device will continue to operate in the **USB** and **BAT** modes.

4.3 EFR32 Reset

The EFR32 Wireless SoC 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 EFR32 will also be issued during Board Controller boot-up. This means that removing power to the Board Controller (plugging out 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 starter kit has a set of peripherals that showcase some of the features of the EFR32.

Be aware that most EFR32 I/O routed to peripherals are also routed to the breakout pads. This must be taken into consideration when using the breakout pads for your application.

5.1 Push Buttons and LEDs

The kit features two user push buttons, marked PB0 (BUTTON0) and PB1 (BUTTON1), and two yellow LEDs, marked LED0 and LED1. The two push buttons are directly connected to the EFR32, and are debounced by a simple RC filter with a time constant of approximately 1 ms. The buttons are connected to pins PF4 and PF5.

The two user LEDs are controlled by the IO-expander on the Radio Board. By default, the IO-expander will drive the LEDs when the button signals are pulled low, so it is possible to reconfigure pins PF4 and PF5 as outputs to control the LEDs. In this case, driving PF4 low will turn on LED0, and driving PF5 low will turn on LED1.

It is also possible to program the LEDs directly by writing to the LED_CTRL (0x03) register in the IO-expander using the I2C bus. If REG_CTRL (bit 7) in LED_CTRL (0x03) is set, the LEDs will indicate the state of bits 0 and 1 of the LED_CTRL register. When this feature is used, the LED state will no longer follow the state of the BUTTON pins.

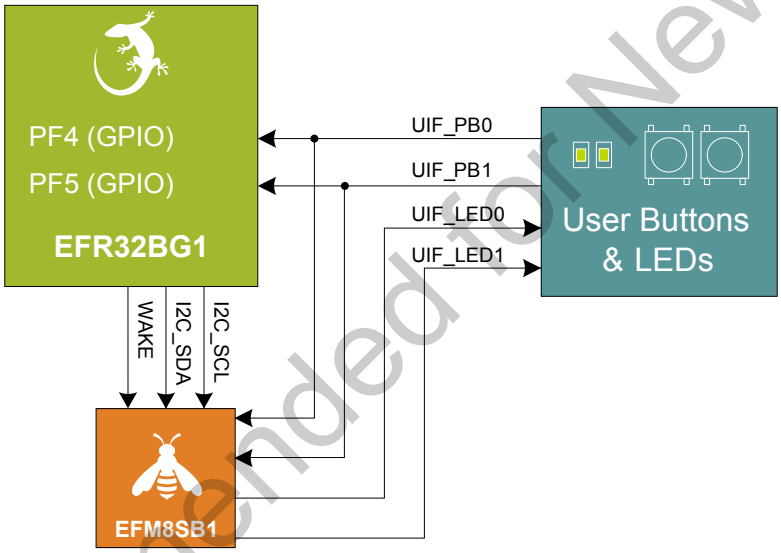

Figure 5.1. Buttons and LEDs

5.2 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 by 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 continous refreshing is required to maintain a static image.

The display interface consists of an 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 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 (EFR32) has control of the display

Power to the display is sourced from the target application power domain when the EFR32 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. Please refer to the display application information for details on driving the display:

http://www.sharpmemorylcd.com/1-28-inch-memory-lcd.html

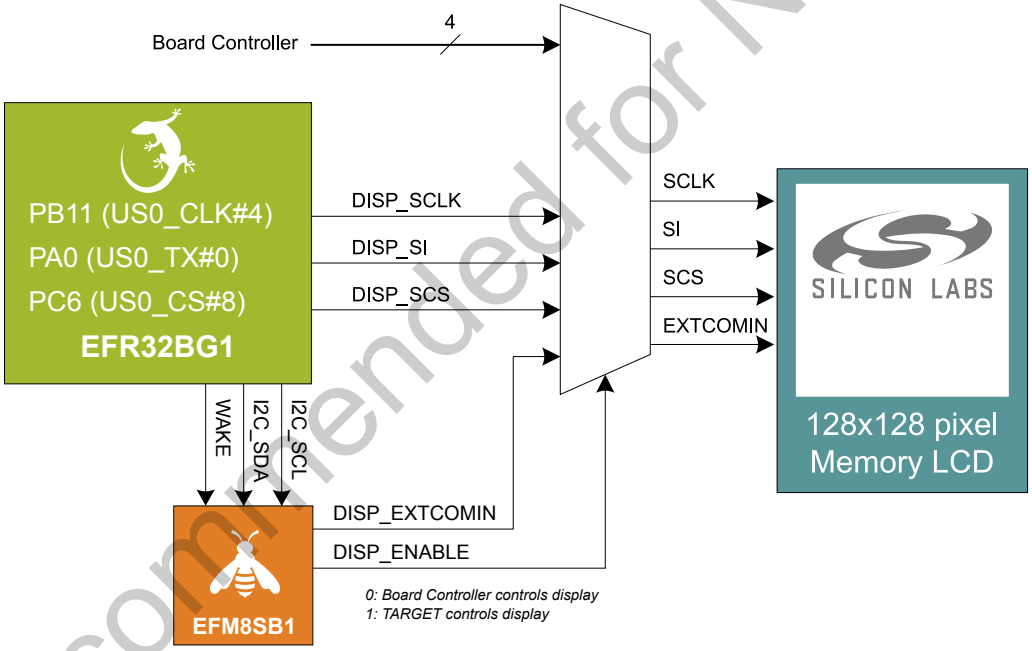

Figure 5.2. 128x128 Pixel Memory LCD

On the BRD4101B Radio Board, both the DISP_ENABLE and the DISP_EXTCOMIN signals are driven by the I^2C I/O expander device. The DISP_ENABLE signal is set by writing to bit 0 (ENABLE) of the DISP_CTRL (0x00) register. The DISP_EXTCOMIN signal is set or cleared by writing to bit 1 (EXTCOMIN) of the DISP_CTRL (0x00) register.

There is also an option to have the I/O expander device automatically handle the periodic toggeling of the DISP_EXTCOMIN signal. If AUTO_EXTCOMIN (bit 2) in DISP_CTRL (0x00) is set, the DISP_EXTCOMIN signal will toggle at a fixed rate of 60 Hz (30 Hz clock output).

Note: The display peripheral shares GPIO pins with the Serial Flash and the Virtual COM port peripherals, as well as the EXP header.

5.3 Serial Flash

The BRD4101B radio board is equipped with an 8 Mbit Macronix MX25R SPI flash that is connected directly to the EFR32. Figure 5.3 Radio Board Serial Flash on page 14 shows how the serial flash is connected to the EFR32.

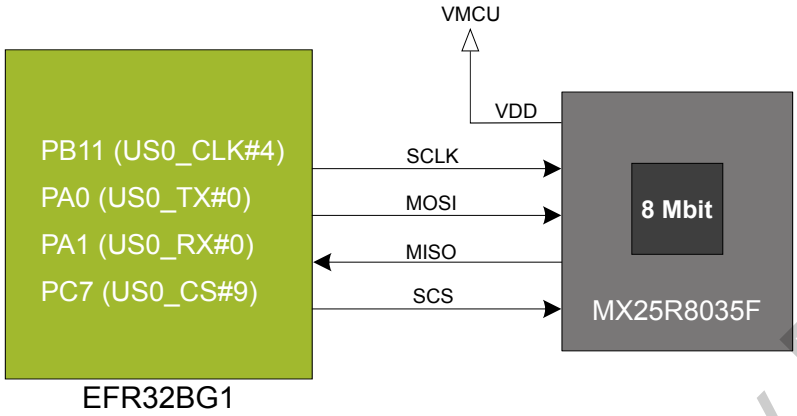

Figure 5.3. Radio Board Serial Flash

The MX25R series are ultra low power serial flash devices, so there is no need for a separate enable switch to keep current consumption down. However, it is important that the flash is always put in deep power down mode when not used. This is done by issuing a command over the SPI interface. In deep power down, the MX25R typically adds approximately 100 nA to the radio board current consumption.

Note: The serial flash peripheral shares GPIO pins with the Display and the Virtual COM port peripherals, as well as the EXP header.

5.4 Si7021 Relative Humidity and Temperature Sensor

The Si7021 I²C 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 I²C 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 I²C bus used for the Si7021 is shared with the Expansion Header. The temperature sensor is normally isolated from the I²C line. To use the sensor, SENSOR_ENABLE must be set high. When enabled, the sensor's current consumption is included in the AEM measurements.

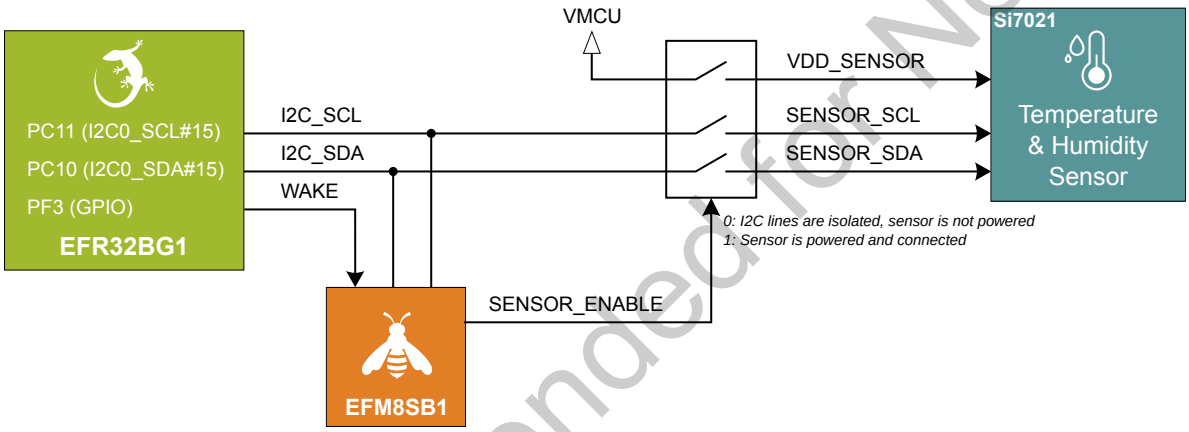

Figure 5.4. Si7021 Relative Humidity and Temperature Sensor

On the BRD4101B Radio Board, the SENSOR_ENABLE signal is driven by the I²C I/O expander device. Enabling the relative humidity & temperature sensor is done by setting ENABLE (bit 0) in the SENSOR_CTRL (0x02) register.

Please 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 EFR32. This eliminates the need for an external serial port adapter.

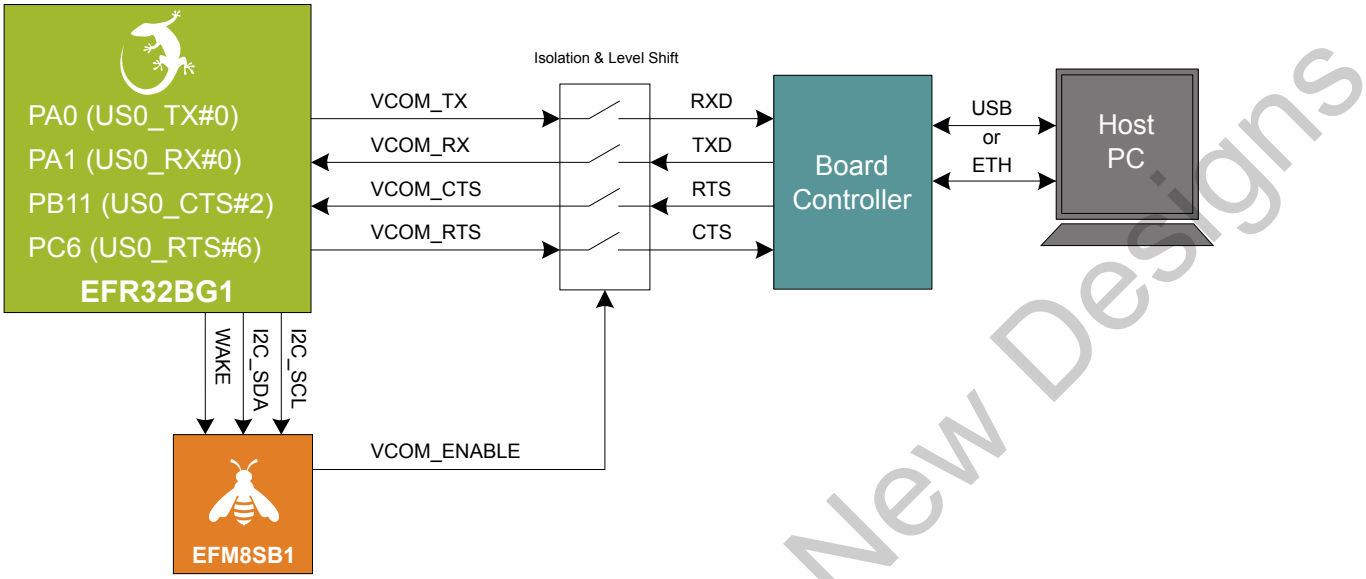

Figure 5.5. Virtual COM Port Interface

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 EFR32 to the board controller
VCOM_RX	Receive data from the board controller to the EFR32
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 EFR32 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 depends on which radio board is used with the Wireless STK Mainboard. Please see for more details.

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

On the BRD4101B Radio Board, the VCOM_ENABLE signal is driven by the I^2C I/O expander device. Enabling the virtual COM port is done by setting ENABLE (bit 0) in the VCOM_CTRL (0x01) register. Note that the Virtual COM Port is enabled by default when the board powers up.

Note: The Virtual COM port shares GPIO pins with the Display and the Serial Flash peripherals, as well as the EXP header.

5.5.1 Host Interfaces

Data sent to the board controller through the VCOM interface is available in two different ways to the user. At the same time, data can be sent to the target device using these interfaces:

- 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. Some examples of device names that can be associated with the kit:

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

Note that these are only examples of what the device might show up as, and the actual assignment depends on the operating system, and how many devices are or have been connected previously. Data sent by the target device into the VCOM interface can be read from this port, and data written to this port is transmitted to the traget 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, with flow control disabled/ignored. (115.2 Kbit/s, 8 databits, 1 stop bit). 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 form 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. Please 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, and 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: Please note that enabling CTS flow control without configuring the VCOM_RTS pin can result in no data being transmitted from the host to the target device.

5.6 I²C I/O expander

The EFR32BG1 2.4 GHz 8 dBm WLCSP radio board features the wafer level chip scale package variant of the EFR32 Blue Gecko Wireless SoC. This variant provides a complete Bluetooth solution in a tiny 3.30 x 3.14 mm package. The package contains 43 BGA balls in a 0.4 mm pitch grid, giving access to a total of 19 General Purpose I/O pins.

Because of the limited number of I/O pins in the WLCSP package, an I²C based I/O expander device was added to the radio board design in order to support advanced Wireless Starter Kit features. This is not part of the general reference design for the EFR32BG1, but rather a Wireless Starter Kit support feature.

The I/O expander off-loads some of the I/O requirements needed to activate certain peripherals of the Wireless Starter Kit, such as:

- The VCOM_ENABLE signal for enabling the virtual COM port interface
- The DISP_ENABLE signal for enabling the Memory LCD-TFT display, as well as the DISP_EXTCOMIN signal required to drive the display.
- The SENSOR_ENABLE signal for enabling the Si7021 Relative Humidity and Temperature Sensor
- The user LEDs

The I/O expander presents a very simple register interface that can be read and written using the I2C bus. It is implemented using a Silicon Labs EFM8 Sleepy Bee device, and consumes very little power during (less than 100 nA) normal operation.

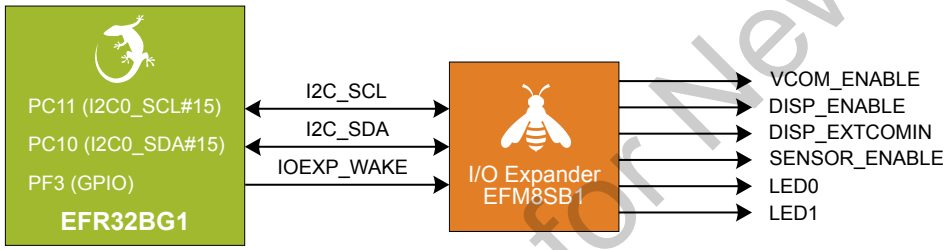

Figure 5.6. I²C I/O Expander (EFM8)

5.6.1 Communication

The EFM8 device that performs the I/O expander function spends most of the time in sleep mode, with its I²C peripheral disabled. In order to perform register accesses to the device, it must first be woken up using the IOEXP_WAKE (PF0) pin. The IOEXP_WAKE pin should be pulled low for the duration of the I²C access. The I²C bus is connected to pins PC10 and PC11 of the EFR32.

Writing to the registers is done by transmittting the 7-bit bus address 0x90 with the R/W bit cleared, followed by the register address and the data to be written. Reading a register first requires the internal address pointer to be set up using an I²C write, followed by a new start together with the address and R/W bit set. Reading can be done with a repeated start, or can be a simple one byte write followed by a read. The figures below illustrate the reading and writing to the I/O expander registers.

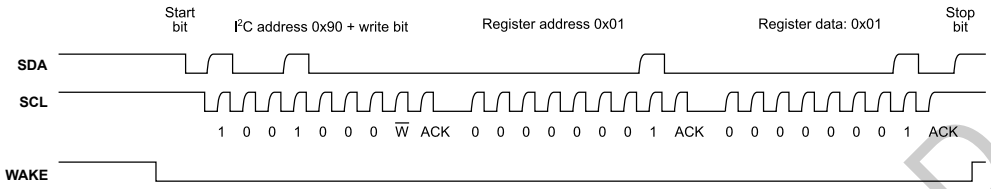

Figure 5.7. I²C Register Write Example

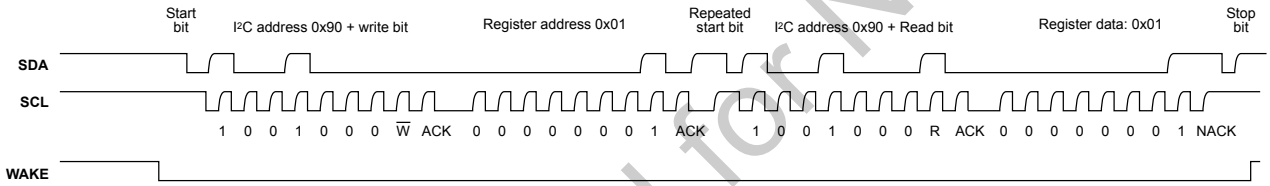

Figure 5.8. I²C Register Read Example

5.6.2 Register Map

The register map and the relevant bit fields of the I/O expander is outlined in the table below.

Table 5.3. I/O Expander Register Map

Addr	Reg Name	Type	Description	Reset
0x00	DISP_CTRL	RW	**Display control register** • Bits 7:3 - Reserved • Bit 2 - AUTO_EXTCOMIN: Enables automatic toggling of the DISP_EXTCOMIN signal at 60 Hz (30 Hz clock output). • Bit 1 - EXTCOMIN: When AUTO_EXTCOMIN is cleared, this bit controls the state of the DISP_EXTCOMIN signal. • Bit 0 - ENABLE: Enables the display by pulling DISP_ENABLE high.	0x00
0x01	VCOM_CTRL	RW	**VCOM enable/disable register** • Bits 7:1 - Reserved • Bit 0 - ENABLE: Enables the virtual COM port interface by pulling VCOM_ENABLE high.	0x01
0x02	SENSOR_CTRL	RW	**Si7021 enable/disable register** • Bits 7:1 - Reserved • Bit 0 - ENABLE: Enables the Si7021 sensor on the Wireless STK by pulling SENSOR_ENABLE high.	0x00

Addr	Reg Name	Type	Description	Reset
0x03	LED_CTRL	RW	**LED control register** • Bit 7 - REG_CTRL: Set this bit to allow control of LED0 and LED1 using this register. If cleared, the LEDs follow the button line states. • Bit 6:2 - Reserved • Bit 1 - LED1: Turns on or off LED1 if REG_CTRL is set. • Bit 0 - LED0: Turns on or off LED0 if REG_CTRL is set.	0x00

6. Board Controller

The Wireless STK Mainboard contains a dedicated microcontroller for some of the advanced kit features provided. This microcontroller is referred to as the "Board Controller", and is not programmable by the user. The board controller acts as an interface between the host PC and the target device on the radio board, as well as handling some house-keeping functions on the board.

Some of the kit features actively managed by the board controller are:

- The On-board Debugger, which can flash and debug both on-board and external targets.
- The Advanced Energy Monitor, which provides real-time energy profiling of the user application.
- The Packet Trace Interface, which is used in conjunction with PC software to provide detailed insight into an active radio network.
- The Virtual COM Port and Virtual UART interfaces, which provide ways to transfer application data between the host PC and the target processor.
- The Admin Console, which provides configuration of the various board features.

Silicon Labs publishes updates to the board controller firmware in form of firmware upgrade packages. These updates may enable new features or fix issues. See 9.1 Firmware Upgrades for details on firmware upgrade.

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 retreiving configuration and operational parameters.

6.1.1 Connecting

The Wireless Starter Kit must be connected to Ethernet using the Ethernet connector in the top left corner of the mainboard for the admin console to be available. See 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 description. 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 interface provides a high performance application data interface that does not require any additional I/O pins apart from the debug interface. It is based on SEGGER's Real Time Transfer (RTT) technology, and uses Serial Wire Output (SWO) to get appliaction data from the device, and a shared memory interface to send data to the target application.

The Wireless Starter Kit makes the Virtual UART interface available on TCP/IP port 4900.

7. Advanced Energy Monitor

7.1 Introduction

Any embedded developer seeking to make his 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. It will in real-time graph and log current as a function of time while correlating this to the actual target application code running on the EFR32. There are multiple features in the profiler software that allows 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 Advanced Energy Monitor (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 alomst 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, and 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 ITM (Instrumentation Trace Macrocell) 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, and automatic AEM calibration is performed. This calibration compensates for any offset errors in the current sense amplifiers.

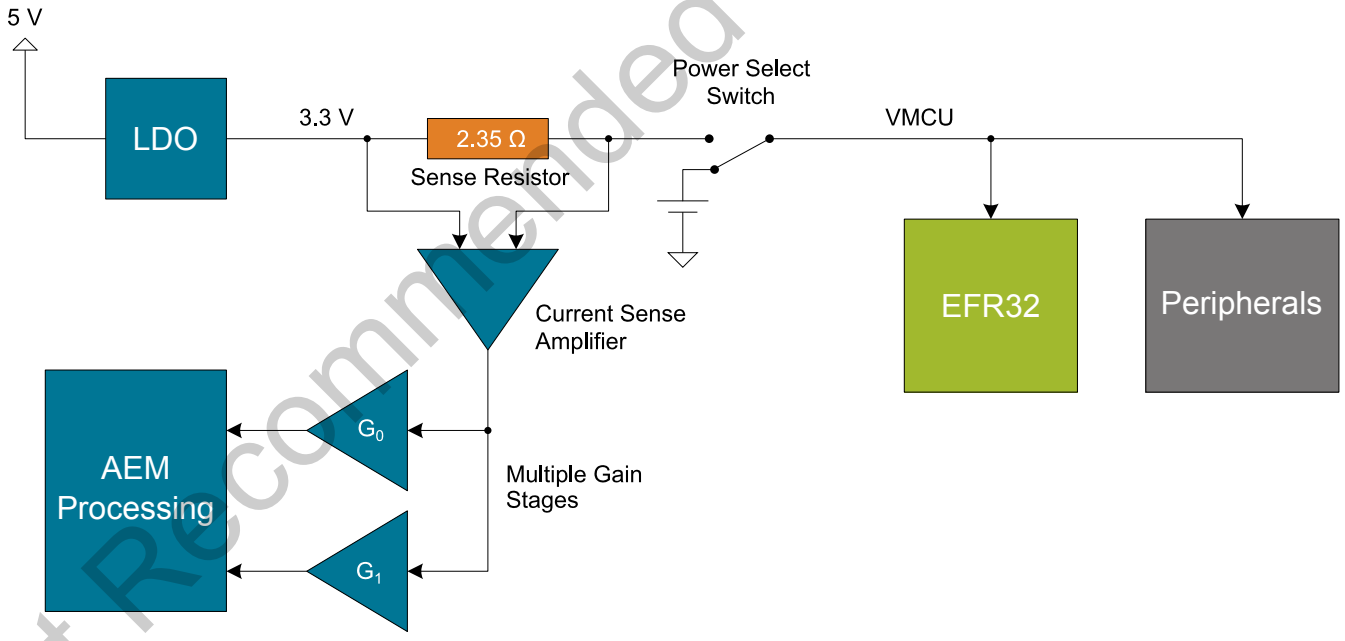

Figure 7.1. Advanced Energy Monitor

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 is able to 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 EFR32 in realtime.

8. On-Board Debugger

The Wireless STK Mainboard contains an integrated debugger, which can be used to download code and debug the EFR32. 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

In order for debugging to work properly, make sure that 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 Starter Kit 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, this typically require 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 Mini-B 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 USB Mini-B connector. See 4.2 Board Controller Power for details.

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

Programming external devices is done by connecting to a target board through the provided Debug IN/OUT Connector, and by setting the debug mode to [**Out**]. The same connector can also be used to connect an external debugger to the EFR32 Wireless SoC 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 EFR32 on the kit.

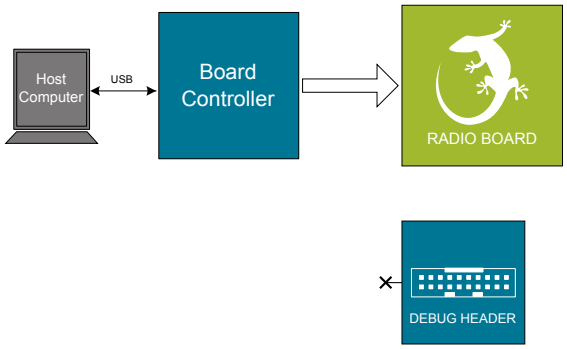

Figure 8.1. Debug MCU

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

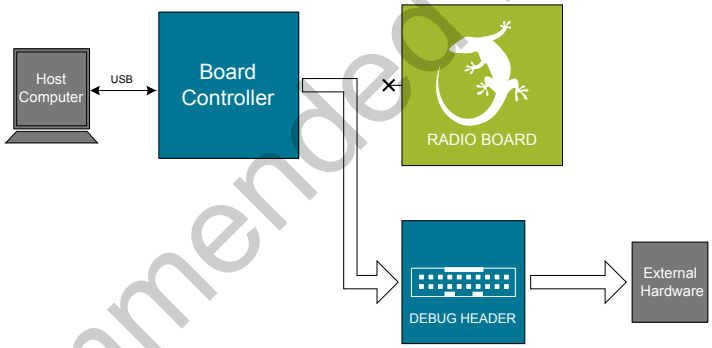

Figure 8.2. Debug OUT

Debug IN: In this mode, the on-board debugger is disconnected, and an external debugger can be connected to debug the EFR32 on the kit.

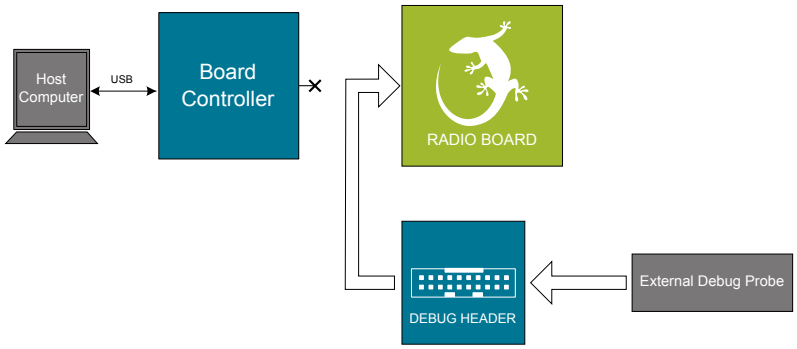

Figure 8.3. Debug IN

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

8.3 Debugging During Battery Operation

When the EFR32 is powered by battery 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 of another energy source, such as a battery, and the board controller is powered down, the user should make direct connections to the GPIO used for debugging. This can be done by connecting to the appropriate pins of the breakout pads. Some Silicon Labs kits provide a dedicated pin header for this purpose.

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.

In the main window of the Simplicity Studio's Launcher perspective, the debug mode and firmware version of the selected J-Link adapter is shown. Click the 'Change' link next to any of them to open the kit configuration dialog.

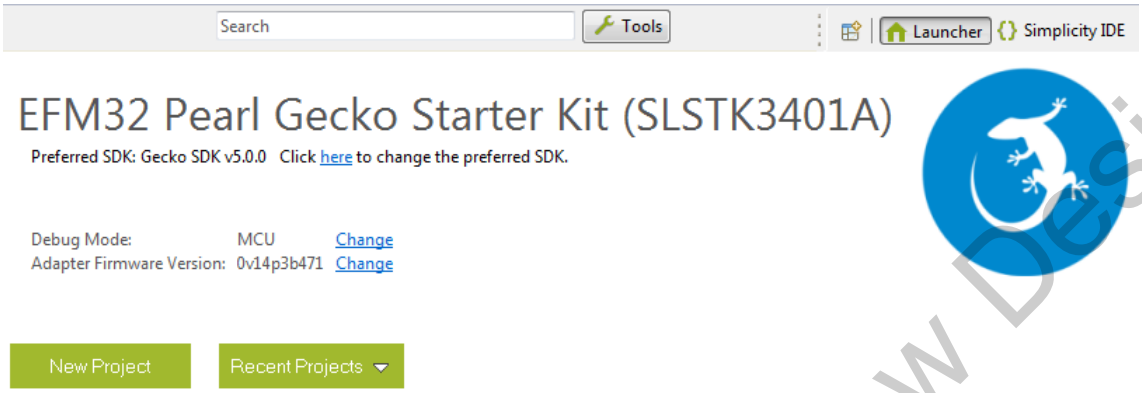

Figure 9.1. Simplicity Studio Kit Information

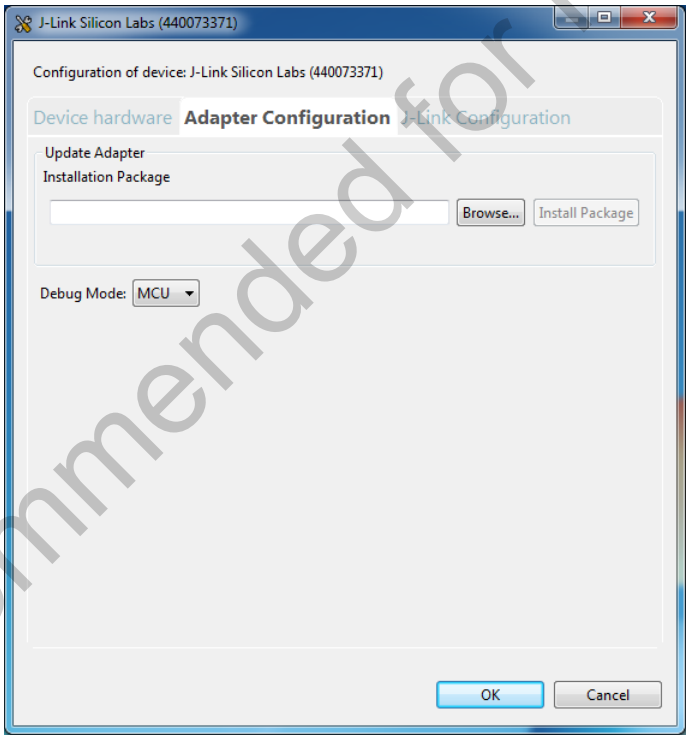

Figure 9.2. Kit Configuration Dialog

9.1 Firmware Upgrades

Upgrading the kit firmware is done 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.

11. Kit Revision History

The kit revision can be found printed on the kit packaging label, as outlined in the figure below.

Figure 11.1. Kit Label

11.1 SLWRB4101B Revision history

Kit Revision	Released	Description
A00	2017-02-08	Initial release.

12. Document Revision History

Revision 1.00

2017-02-21

Initial document version.

Table of Contents

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 and Community
community.silabs.com

SILICON LABS

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

http://www.silabs.com