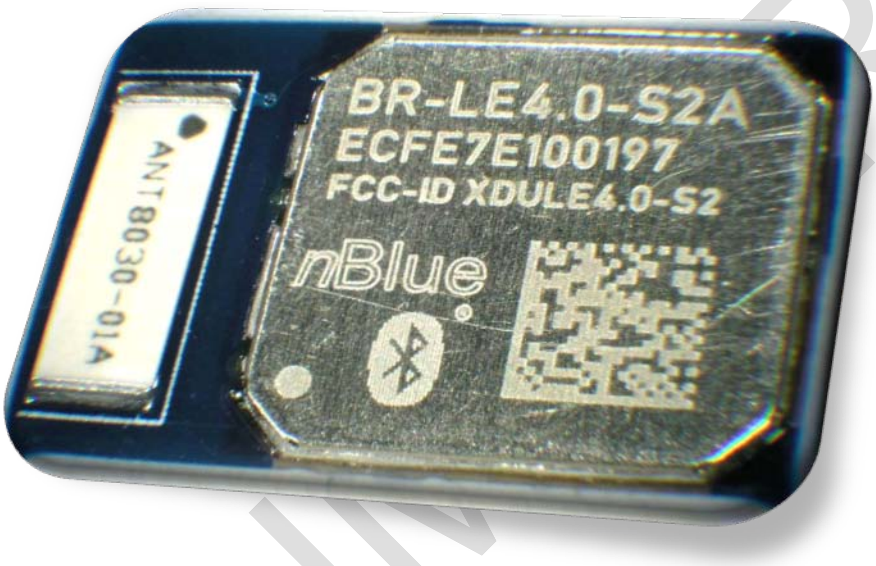
*n*Blue™ *Bluetooth*® 4.0
Module User's Guide
V3.1.0



BR-LE4.0-S2A Single Mode Low Energy Module
(Actual Size Not Shown)

AT HOME. AT WORK. ON THE ROAD. USING BLUETOOTH WIRELESS TECHNOLOGY MEANS TOTAL FREEDOM FROM THE CONSTRAINTS AND CLUTTER OF WIRES IN YOUR LIFE.

Table of Contents

Revision History

Rev #	Date	Description
2.0.0	4/13/2012	**Initial Document** ▪ AT commands have been separated from the User's Guide into a separate AT.s Command Set document, which now includes commands for dual mode modules as well as single mode modules.
3.1.0	5/12/2012	▪ Added Dual Mode BR-LE4.0-D2A module hardware details

1 Introduction

"Our clients buy our products because they are reliable and easy to integrate, enabling them to quickly deploy cost-effective wireless solutions."

Mark J. Kramer – CEO of ***Blue*****Radios**

1.1 Scope

This document along with the ***Blue*****Radios®** ***n*****Blue™** *Bluetooth®* low energy (BLE) evaluation kit was created to enable developers and integrators an opportunity to evaluate wireless networks using BLE technology. The goal is to make the transition to BLE as seamless and as easy as possible for our clients. This document will explain the basics of BLE communications and provide module hardware details.

1.2 Background

Bluetooth low energy was designed to enable the development of low complexity, low cost wireless devices that require minimal power consumption, such as sensors and watches. These devices typically transmit very small data packets at a time, while consuming as little power as possible. *Bluetooth* Version 4.0 specifies two types of implementation for BLE devices: single-mode and dual-mode. Single-mode chips implement the low energy specification and consume just a fraction of the power of classic *Bluetooth* (BR/EDR), allowing the short-range wireless standard to extend to coin cell battery applications. Dual mode chips combine low energy with the power of classic *Bluetooth* and are likely to become a standard feature in almost all new *Bluetooth* enabled cellular phones and computers (i.e., gateway devices).

The ***Blue*****Radios** ***n*****Blue™** modules are *Bluetooth* Version 4.0 compliant. The modules are designed to be built into an embedded device and to provide a simple, reliable, and low cost API interface. The module is designed to integrate with a wide range of applications and platforms.

2 Important Notes – Please Read Prior To Continuing

2.1 Pin Voltage Levels

- The maximum voltage level on any pin should not exceed 3.6V. **The I/O is NOT 5V tolerant.**

2.2 Firmware Upgrades

- Firmware upgrades should only be performed using the nBlue™ Programmer, not a CC Debugger. After BlueRadios programs the bootloader into the modules the debug interface is locked. In order to program an nBlue™ module using a Debugger it will then need to be unlocked, which will erase the entire flash including the module's bootloader, making it incapable of performing firmware updates using the nBlue™ programmer. Once the bootloader has been erased it cannot be reinstalled in the field, it is a factory process only.

- However, custom software can be flashed using the nBlue™ programmer, see the Firmware Upgrades section for more information.

2.3 TI Software

- Firmware from TI can be used on BR-LE4.0-S2A and BR-LE4.0-D2A modules, but this requires erasing the entire flash including the module's bootloader, making it incapable of performing firmware updates using the nBlue™ programmer. Once the bootloader has been erased it cannot be reinstalled in the field, it is a factory process only.

2.4 Related Documents

- nBlue AT.s Command Set

- nBlue BR-EVAL-4.0-X2A Quick Start Guide

3 BLE Basics

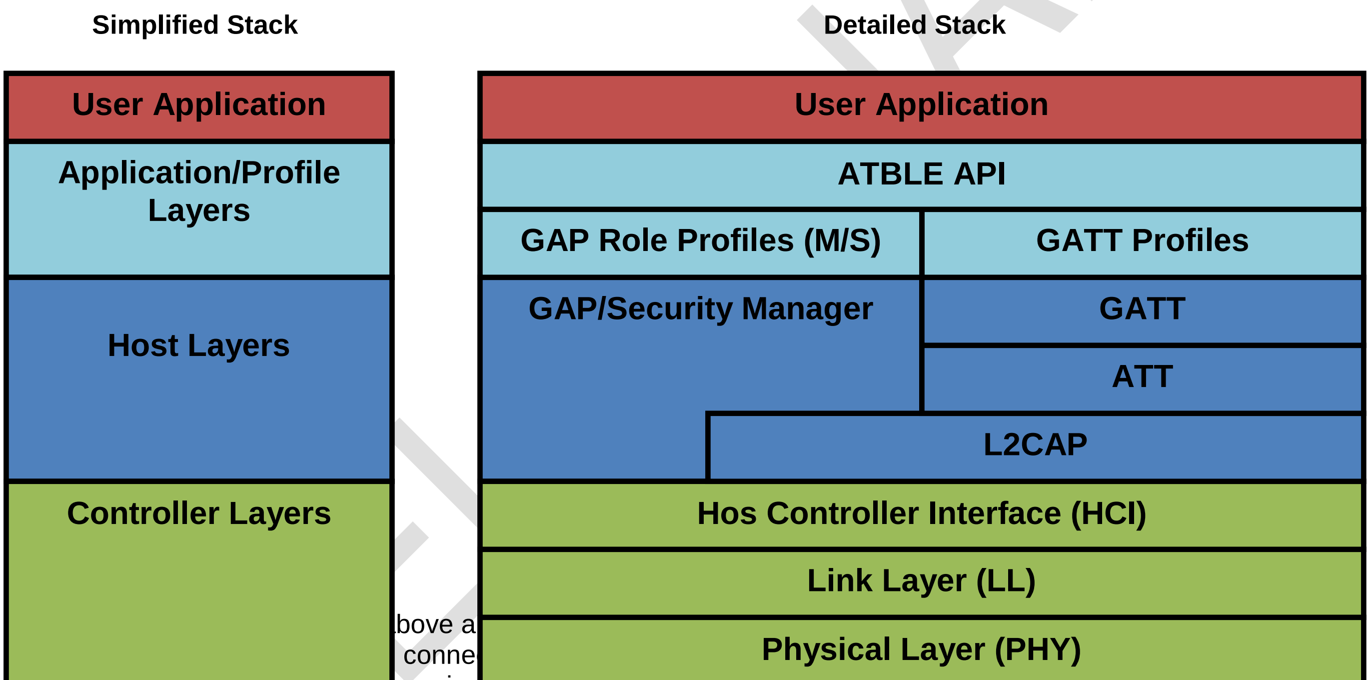
This section is designed to give the user a basic overview of Bluetooth Low Energy.

3.1 Radio Specifications

- 1 Mbps Adaptive Frequency-Hopping GFSK 2.4GHz ISM Radio
- 40 physical channels total, 37 for data, 3 for advertising
- TDMA polling scheme used to exchange data

3.2 Protocol Stack

The *Bluetooth* low energy protocol stack is built similarly to BR/EDR stack, with a controller layer at the bottom, followed by a host layer, and finally the profile and application layers. The ***n*Blue™** modules contain a full stack all the way up to the application layer as seen below. The ATBLE API contains all the necessary commands to enable the user to quickly and easily develop a fully-featured BLE application. The User Application can use the ATBLE API through the UART, using an external microcontroller, or by writing a custom application to run directly on the module.



bove a

conne

The GATT layer handles the exchanging of data. AT commands for directly using the GATT layer will be added in a future release.

3.3 Using BLE

Bluetooth low energy was designed to enable the development of low complexity, low cost devices that require minimal power consumption. These devices typically transmit very small data packets at a time, while consuming as little power as possible.

3.3.1 Typical Use Case

In a typical use case, a slave device will advertise itself as a connectable device. It will send out advertising packets consisting of its *Bluetooth* Device Address as well as other useful information. A master device in the discovery state can receive this information, and if it desires do a scan request to ask the slave for more data, such as its *Bluetooth* Device Name. With the information it has received the master can initiate a connection with the slave. Once connected, the two devices can exchange data bi-directionally through the GATT layer.

3.3.2 BlueRadios Serial Port Profile (BRSP)

The ***n*Blue™** modules use a proprietary GATT profile developed by BlueRadios to stream data; it is not an official *Bluetooth* profile. BlueRadios serial port implementation simplifies the user experience, allowing users to stream data similar to the way the official *Bluetooth* Serial Port Profile (SPP) works on BR/EDR devices. It allows the ***n*Blue™** modules to behave very similar to the BlueRadios ATMP modules. BlueRadios will implement standard profiles as they are defined and published by the *Bluetooth* SIG.

Upon making a connection, both master and slave modules will search the GATT services of the other device for the BRSP profile. If it is found the devices will exchange some data to setup a data stream, and then both will enter data mode.

3.3.3 Connection Parameters

The connection parameters are the four most important parameters of any BLE device. They control the how often two devices in a connection will meet up to exchange data in a connection event, which means they control the balance between throughput and power savings.

- **Connection Interval –** The amount of time between two connection events. A shorter connection interval will result in higher throughput, lower latency, but will use more power, and a longer connection interval will result in lower throughput, higher latency and less power consumption.

- **Slave Latency –** Allows a slave to skip a set number of connection events. This allows the slave to save power if it has no data to send. This will not affect the latency of data sent from slave to master, but will result in increased latency from master to slave. The slave latency must not make the effective connection interval greater than 32s.

- **Supervision Timeout –** Amount of time allowed since the last successful connection event. If this timeout occurs the connection will be dropped. The supervision timeout must not be greater than the effective connection interval.

- **Effective Connection Interval –** Amount of time between connection events, assuming the maximum connection interval is used and the slave always skips [slave latency] connection events. It can be calculated with the following formula:

  Effective Connection Interval **=** *(Max Connection Interval) * (1+(Slave Latency))*

3.3.4 Advertising/Discovery

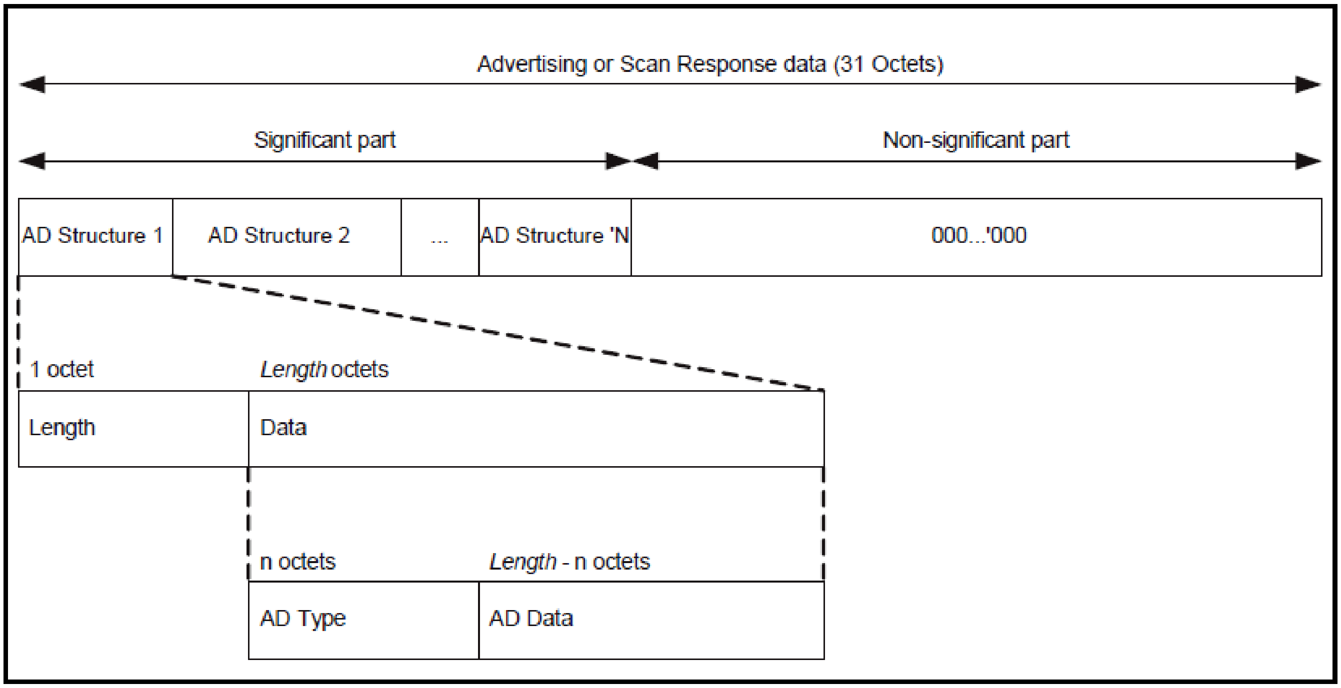
- **Active vs. Passive Mode –** A master in the discovering state will receive advertising packets from slaves in the advertising state. In a passive scan the master will only print the data contained in these packets. In an active scan, after receiving an advertising packet, the master will issue a scan request and ask the slave for additional data. The slave will then respond with a scan response which will be printed by the master.

- **Filtering –** The filter allows a master to filter slave advertising packets based on the type of advertiser the slave is. Slaves in the limited discoverable mode are only discoverable for a limited amount of time, for example a device that requires a button hold to make it discoverable. By default the BlueRadios slave modules are always in the general discoverable mode when discoverable, *this will be modifiable in a future release.*

- **AD Types –** There are four different types of advertisements that a slave can send out. By default the BlueRadios slave modules always send connectable undirected advertisements, *this will be modifiable in a future release.*

0. **Connectable undirected advertisement –** This type of advertisement can be received by any master device, and lets the master know that the slave is connectable.
1. **Connectable directed advertisement –** This type of advertisement can only be received by a specific master device, and lets the master know that the slave is connectable.
2. **Scannable undirected advertisement –** This type of advertisement can be received by any master device, and lets the master know that the slave will accept scan requests, but is not connectable.
3. **Non-connectable undirected advertisement –** This type of advertisement can be received by any master device, and lets the master know that the slave is not connectable and does not accept scan requests.

- **AD Structures –** Advertising packets are made up of multiple ad structures, as seen in the figure below. Each one consists of a different type of data which is defined by the AD Structure Type. Descriptions of the different AD Structure Types can be found in Appendix B.

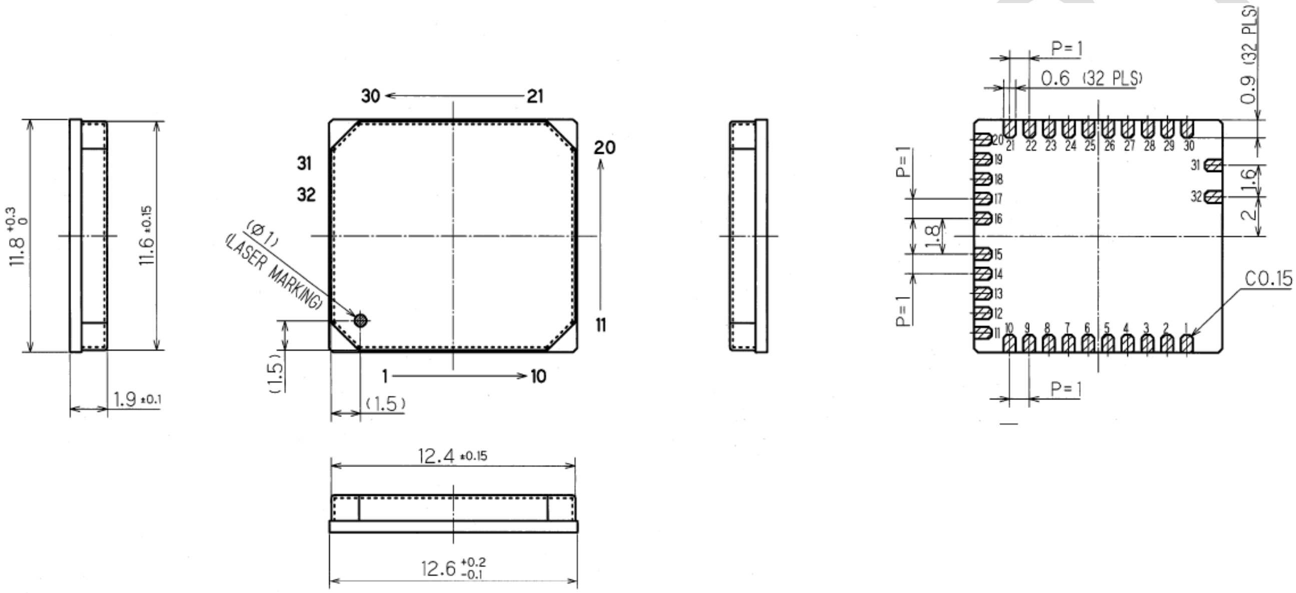


4 Module Hardware Details

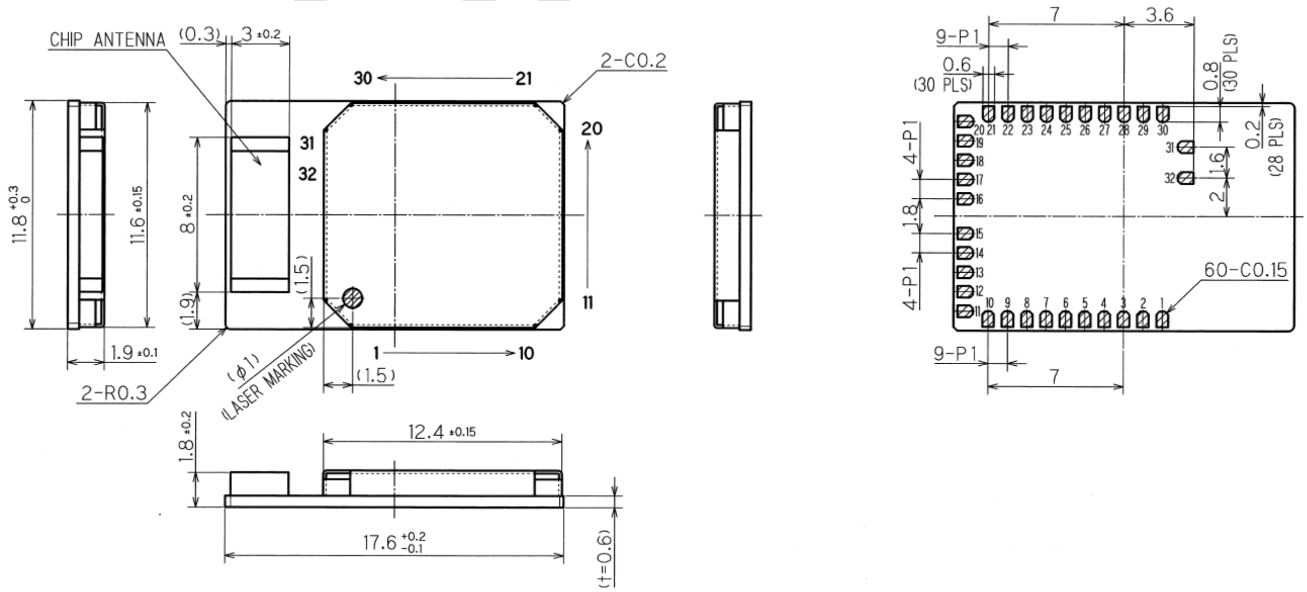
The ***Blue*****Radios** BLE modules were designed to be as similar as possible to the ***Blue*****Radios** *Bluetooth* Version 2.0 modules. The footprint of the BR-LE4.0-S2 and BR-LE4.0-D2 is identical to the C46 module, with some small differences in the pin out.

4.1 BR-LE4.0-S2/BR-LE4.0-D2 Common Features

4.1.1 BR-LE4.0-X2N Dimensions (No Antenna)

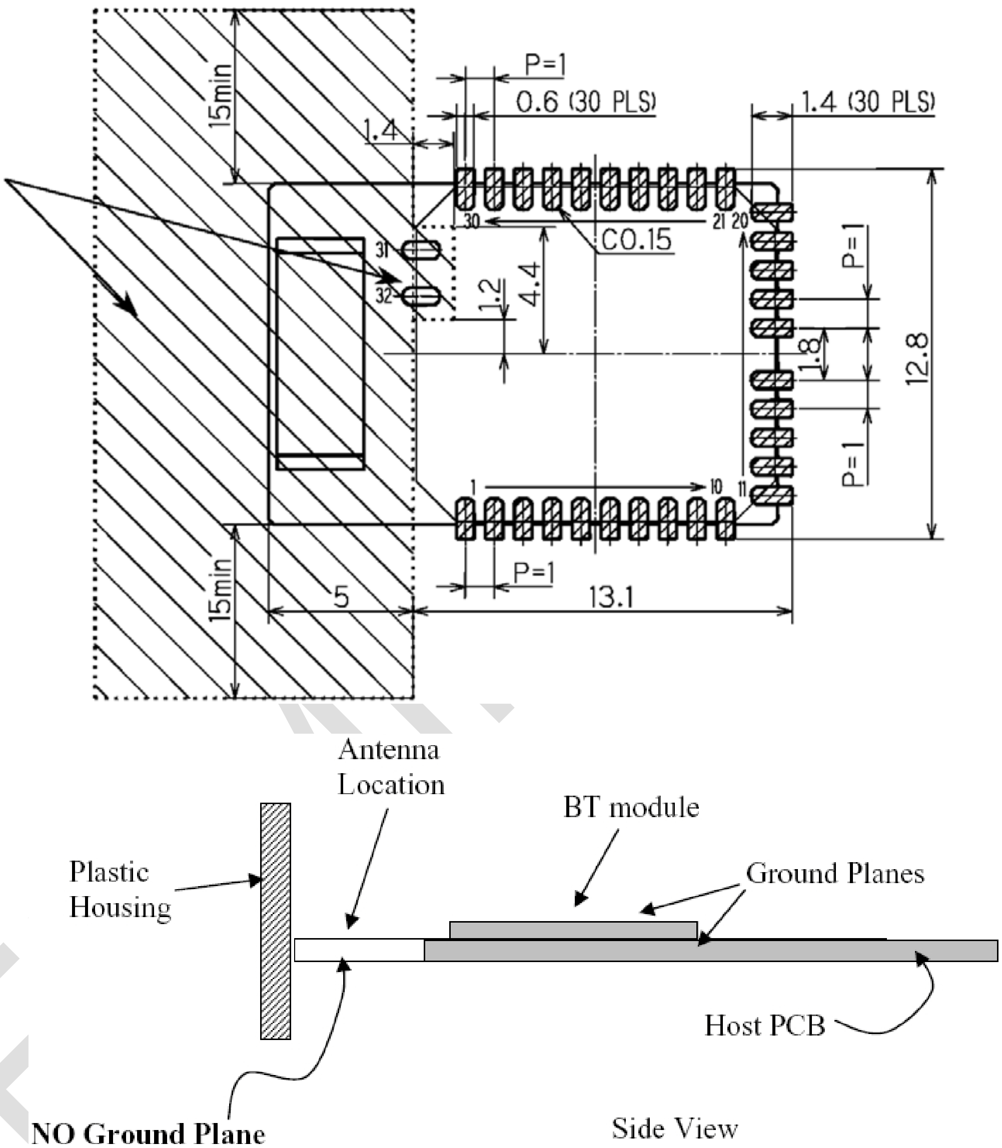


4.1.2 BR-LE4.0-X2A Dimensions (With Antenna)



4.1.3 BR-LE4.0-X2A Standard Land Dimensions (With Antenna)

In this area, you should not locate any parts or GND plane / Patterns on surface or internal layer.



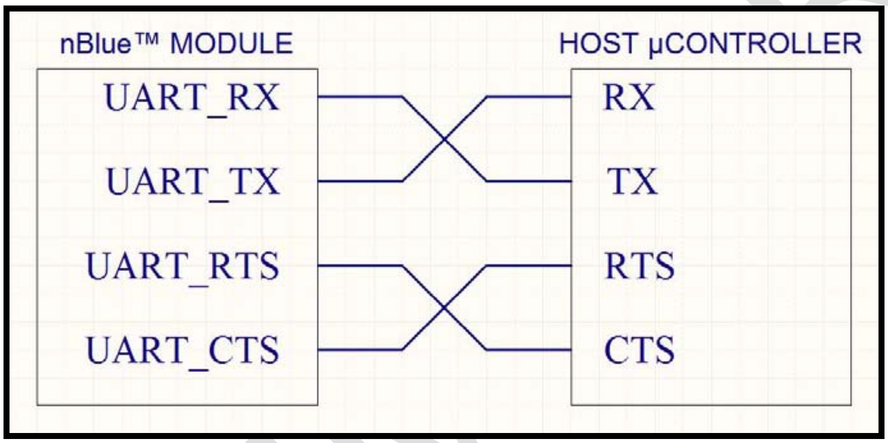
Note: Radio requires a RF ground plane on the rest of the Printed Circuit Board (PCB) area. This can be located on any layer of the PCB. Extend the RF ground plane parallel to module pins 31 and 32 the entire length of your board. Connect all ground pins and do not notch the ground plane around the module. Bottom of module is grounded so be careful of vias or conductive traces located under the modules that are not soldered masked to prevent shorting. Keep metallic components, connectors, copper traces, internal layers, and ground planes away from the antenna area in 3D space!

4.2 Power-up and Reset

There are no strict requirements for power up timing. To reset the module, the RESET line must be pulsed low for at least 1µS.

4.3 UART Interface

UART_TX, UART_RX, UART_RTS and UART_CTS form a conventional asynchronous serial data port. Two-way hardware flow control is implemented by UART_RTS and UART_CTS. These signals operate according to normal industry convention. The signaling levels are nominal 0V and VDD and are inverted with respect to the signaling on an RS232 cable.



4.4 PIO_14 Firmware Upgrade Mode

PIO_14 can be used to manually put the module into firmware upgrade mode by setting it to VDD during power up or reset and holding it at VDD until both PIO_2 and PIO_5 output VDD.

4.5 BR-LE4.0-S2A

The BR-LE4.0-S2A utilizes the Texas Instruments CC2540 SoC. For detailed specifications see the CC2540 datasheet: http://focus.ti.com/lit/ds/symlink/cc2540.pdf

4.5.1 Electrical Specifications Summary

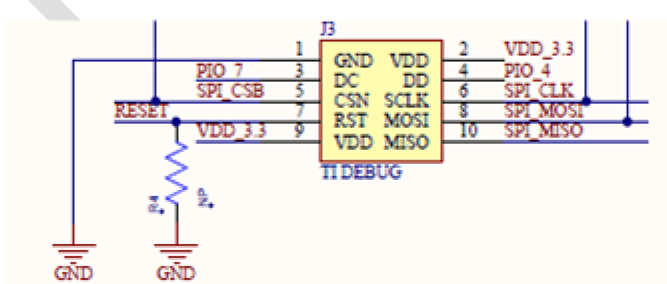
- The modules operate at a supply voltage (VDD) of 2.0-3.6V, 3.0V is recommended.
- VDD ripple should not exceed 100mV.
- Minimum logic high input voltage is 2.5V
- Maximum voltage level on any pin should not exceed 3.6V. **The I/O is not 5V tolerant.**
- All PIOs have a 4mA drive capability, except for PIO_2 and PIO_5, which can drive 20mA.
- Applying VDD to a PIO set to an output may permanently damage the module.

4.5.2 RF Specifications Summary

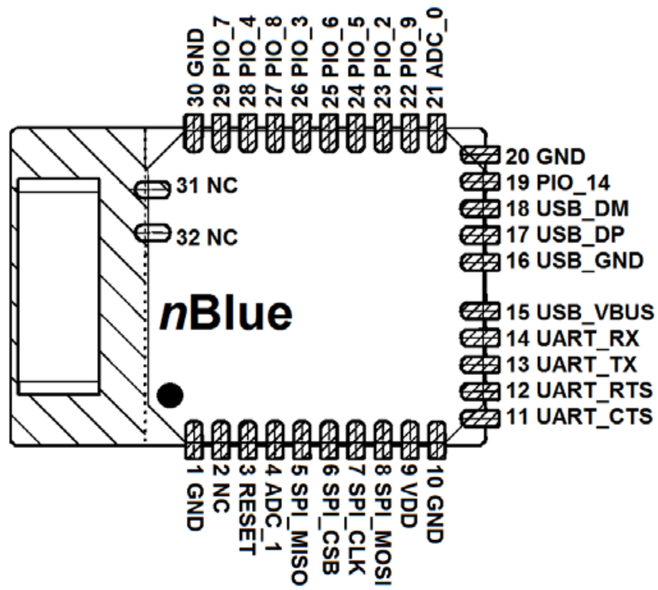
Item	Specifications
Frequency	2402 - 2480MHz
Modulation	GFSK, 250 KHz deviation
Channel intervals	Programmable 1 MHz steps
Number of channels	40CH: 37 AFH data channels. 3 discoverable, connectable and broadcasting channel
Power supply voltage	2.0 to 3.6Vdc and < 10mVp-p noise
Current consumption	22mA @4dBm
Transmission rate (over the air)	1Mbps
Receive sensitivity (w/chip antenna)	-95dBm typ.
Output Power (Class2)	4dBm max.
RX/TX turnaround	150 usec. with 99dBm link budget

4.6 Debugging

PIO_4 and PIO_7 also function as the Debug Data (DD) and Debug Clock (DC) lines, allowing the modules to be connected to a TI CC Debugger for debugging and programming. The debug headers on the on the evaluation boards can be connected directly to a CC Debugger. **This functionality is only needed if you are writing a custom application for a module and not using the ATBLE commands.** See the CC Debugger User's Guide for more information: http://focus.ti.com/lit/ug/swru197c/swru197c.pdf



4.7 BR-LE4.0-S2 Pin Out



Pin	Pin Name	I/O Dir.	Description (Default/Optional)
1	GND	-	Ground
2	NC	-	No Connect
3	RESET	IN	Active-low reset
4	ADC_1	IN/OUT	PIO_1/ ADC Input 1
5	SPI_MISO	IN/OUT	PIO_10 / SPI Master In Slave Out / ADC Input 2
6	SPI_CSB	IN/OUT	PIO_11/ SPI Chip Select / ADC Input 4
7	SPI_CLK	IN/OUT	PIO_12 / SPI Clock / ADC Input 5
8	SPI_MOSI	IN/OUT	PIO_13 / SPI Master Out Slave In / ADC Input 3
9	VDD	-	2.0-3.6V
10	GND	-	Ground
11	UART_CTS	IN	UART CTS / PIO_15
12	UART_RTS	OUT	UART RTS / PIO_16
13	UART_TX	OUT	UART Transmit / PIO_17
14	UART_RX	IN	UART Receive / PIO_18
15	USB_VBUS	-	-
16	USB_GND	-	-
17	USB_DP	-	-
18	USB_DM	-	-
19	PIO_14	IN/OUT	PIO_14 / ADC Trigger / Firmware Upgrade on Reset
20	GND	-	Ground
21	ADC_0	IN/OUT	PIO_0 / ADC Input 0
22	PIO_9	IN/OUT	PIO_9
23	PIO_2	OUT	PIO_2
24	PIO_5	OUT	PIO_5
25	PIO_6	IN/OUT	PIO_6 / ADC Input 7
26	PIO_3	IN	PIO_3
27	PIO_8	IN/OUT	PIO_8
28	PIO_4	IN	PIO_4
29	PIO_7	IN	PIO_7 / DC

30	GND	-	Ground
31	NC	-	RF Test Antenna
32	NC	-	RF Test Ground

4.7.1 BR-LE4.0-S2 to CC2540 I/O Mapping

BR-LE4.0-S2 Pin	BR-LE4.0-S2 Pin Name	CC2540 Pin Name
4	ADC_1	P0_1
5	SPI_MISO	P0_2
6	SPI_CSB	P0_4
7	SPI_CLK	P0_5
8	SPI_MOSI	P0_3
11	UART_CTS	P1_4
12	UART_RTS	P1_5
13	UART_TX	P1_6
14	UART_RX	P1_7
15	USB_VBUS	DVDD_USB
16	USB_GND	DGND_USB
17	USB_DP	USB_P
18	USB_DM	USB_N
19	PIO_14	P2_0
21	ADC_0	P0_0
22	PIO_9	P1_3
23	PIO_2	P1_0
24	PIO_5	P1_1
25	PIO_6	P0_7
26	PIO_3	P0_6
27	PIO_8	P1_2
28	PIO_4	P2_1
29	PIO_7	P2_2

4.8 BR-LE4.0-D2A

The BR-LE4.0-D2A utilizes the Texas Instruments MSP4305438A microcontroller . For detailed specifications see the MSP4305438A datasheet: http://www.ti.com/lit/ds/symlink/msp430f5438a.pdf

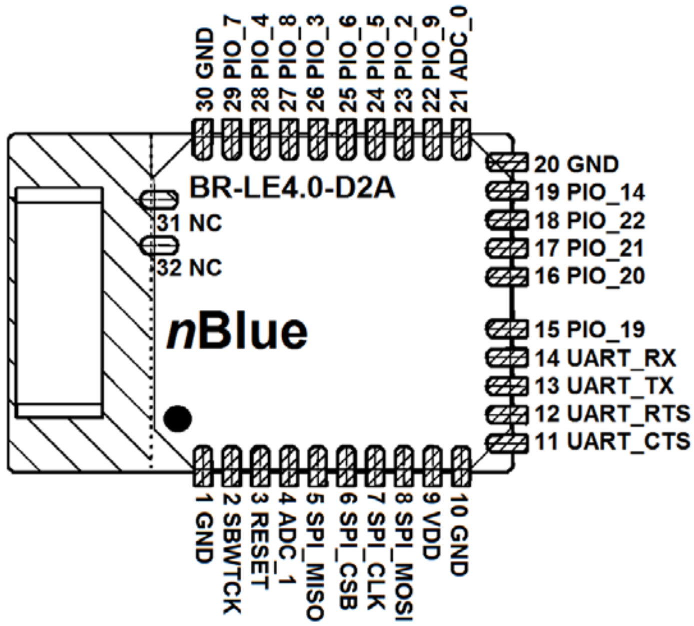
4.8.1 Electrical Specifications Summary

- The modules operate at a supply voltage (VDD) of 2.0-3.6V, 3.0V is recommended.
- VDD ripple should not exceed 100mV.
- Minimum logic high input voltage is 2.5V
- Maximum voltage level on any pin should not exceed 3.6V. **The I/O is not 5V tolerant.**
- All PIOs have a 4mA drive capability, except for PIO_2 and PIO_5, which can drive 20mA.
- Applying VDD to a PIO set to an output may permanently damage the module.

4.8.2 RF Specifications Summary

Item	Specifications
Frequency	2402 - 2480MHz
Modulation	GFSK, 250 KHz deviation
Channel intervals	Programmable 1 MHz steps
Number of channels	40CH: 37 AFH data channels. 3 discoverable, connectable and broadcasting channel
Power supply voltage	2.0 to 3.6Vdc and < 10mVp-p noise
Current consumption	22mA @4dBm
Transmission rate (over the air)	1Mbps
Receive sensitivity (w/chip antenna)	-95dBm typ.
Output Power (Class2)	4dBm max.
RX/TX turnaround	150 usec. with 99dBm link budget

4.9 BR-LE4.0-D2 Pin Out



Pin	BR-LE4.0-D2 Pin Name	MSP430F5438A Pin Name	Description/ Alt Functions
1	GND		
2	TEST/SBWTCK	TEST/SBWTCK	Test mode pin – Selects four wire JTAG operation, Spy-Bi-Wire input clock when Spy-Bi-Wire operation activated
3	RESET	RST	Reset input active low, I/O Non-maskable interrupt input, Spy-Bi-Wire data input/output when Spy-Bi-Wire operation activated.
4	ADC 1 (PIO1)	P6.1	General-purpose digital I/O, Analog input A1 – ADC
5	SPI MISO (PIO 10)	P3.2	General-purpose digital I/O, Slave out, master in – USCI_B0 SPI mode, I2C clock – USCI_B0 I2C mode
6	SPI CS (PIO 11)	P3.0	General-purpose digital I/O, Slave transmit enable – USCI_B0 SPI mode
7	SPI CLK (PIO 12)	P3.3	General-purpose digital I/O, Clock signal input – USCI_B0 SPI slave mode Clock signal output – USCI_B0 SPI master mode
8	SPI MOSI (PIO 13)	P3.1	General-purpose digital I/O, Slave in, master out – USCI_B0 SPI mode, I2C data – USCI_B0 I2C mode
9	VDD		2.0V-3.9V
10	GND		

11	UART0 CTS (PIO 15)	P2.1	General-purpose digital I/O with port interrupt, TA1 CCR0 capture: CCI0A input, compare: Out0 output
12	UART0 RTS (PIO 16)	P2.0	General-purpose digital I/O with port interrupt, TA1 clock signal TA1CLK input, MCLK output
13	UART0 TX (PIO 17)	P10.4	General-purpose digital I/O, Transmit data – USCI_A3 UART mode,
14	UART0 RX (PIO 18)	P10.5	General-purpose digital I/O, Receive data – USCI_A3 UART mode,
15	PIO 19	P5.6	General-purpose digital I/O, Transmit data – USCI_A1 UART mode, Slave in, master out – USCI_A1 SPI mode
16	PIO 20	P5.7	General-purpose digital I/O, Receive data – USCI_A1 UART mode, Slave out, master in – USCI_A1 SPI mode
17	PIO 21	P3.6	General-purpose digital I/O, Clock signal input – USCI_A1 SPI slave mode, Clock signal output – USCI_A1 SPI master mode
18	PIO 22	P5.5	General-purpose digital I/O, Slave transmit enable – USCI_A1 SPI mode
19	PIO 14	P6.6	General-purpose digital I/O, Analog input A6 – ADC
20	GND		
21	ADC 0 (PIO 0)	P6.0	General-purpose digital I/O, Analog input A0 – ADC
22	PIO 9	P5.0	General-purpose digital I/O, Analog input A8 – ADC, Output of reference voltage to the ADC, Input for an external reference voltage to the ADC
23	PIO 2	P1.1	General-purpose digital I/O with port interrupt, TA0 CCR0 capture: CCI0A input, compare: Out0 output, BSL transmit output
24	PIO 5	P1.5	General-purpose digital I/O with port interrupt, TA0 CCR4 capture: CCI4A input, compare: Out4 output
25	PIO 6	P1.6	General-purpose digital I/O with port interrupt, SMCLK output
26	PIO 3	P1.0	General-purpose digital I/O with port interrupt, TA0 clock signal TACLK input, ACLK output (divided by 1, 2, 4, or 8)
27	PIO 8	P5.1	General-purpose digital I/O, Analog input A9 – ADC, Negative terminal for the ADC's reference voltage for both sources, the internal reference voltage, or an external applied reference voltage

28	PIO 4	P1.2	General-purpose digital I/O with port interrupt, TA0 CCR1 capture: CCI1A input, compare: Out1 output, BSL receive input
29	PIO 7	P6.7	General-purpose digital I/O,Analog input A7 – ADC
30	GND		
31	ANT		
32	GND		

5 Evaluation Boards

5.1 Drivers

The Dongle and the DevBoard both require virtual COM port drivers to work with a host OS, which can be obtained from: http://www.ftdichip.com/Drivers/VCP.htm. Most OS's will automatically locate and install these drivers.

5.2 Debuggers

5.2.1 CC Debugger for BR-LE4.0-S2

When using a CC Debugger with any of the *n*Blue™ evaluation boards, be sure that only one power source is supplying the board at a time. Either disconnect the board from its power source before connecting the CC Debugger, or to enable debugging without disconnecting from your power source, disconnect pin 9 on the debug header by cutting wire 9 on the CC Debugger cable.

5.2.2 MSP-FET430UIF Debugger for BR-LE4.0-D2

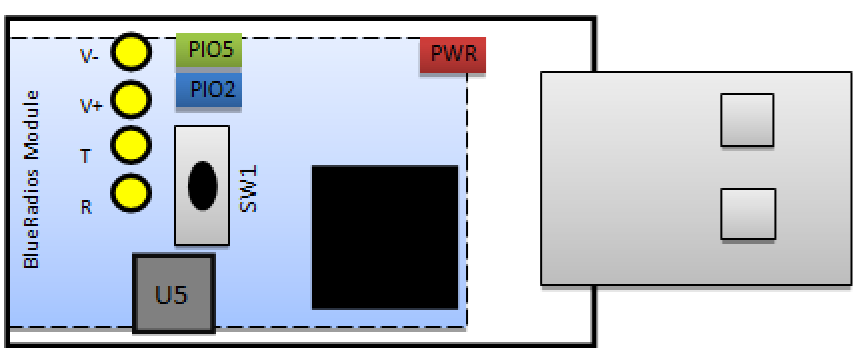
When using a Texas Instuments MSP430 USB-Debug-Interface MSP-FET430UIF Debugger with any of the *n*Blue™ evaluation boards, be sure to only use the MSP-FET430UIF Debugger power source.

5.3 *n*Blue™ BR-MUSB-LE4.0-X2A Mini Dongle

This Dongle provides users with a smaller, ready to use Single or Dual Mode Device. The module's UART is connected to a USB to Serial Converter, LEDs are connected to PIOs 2 and 5. If needed, the firmware can be updated by using the nBlueProg program.

Layout:

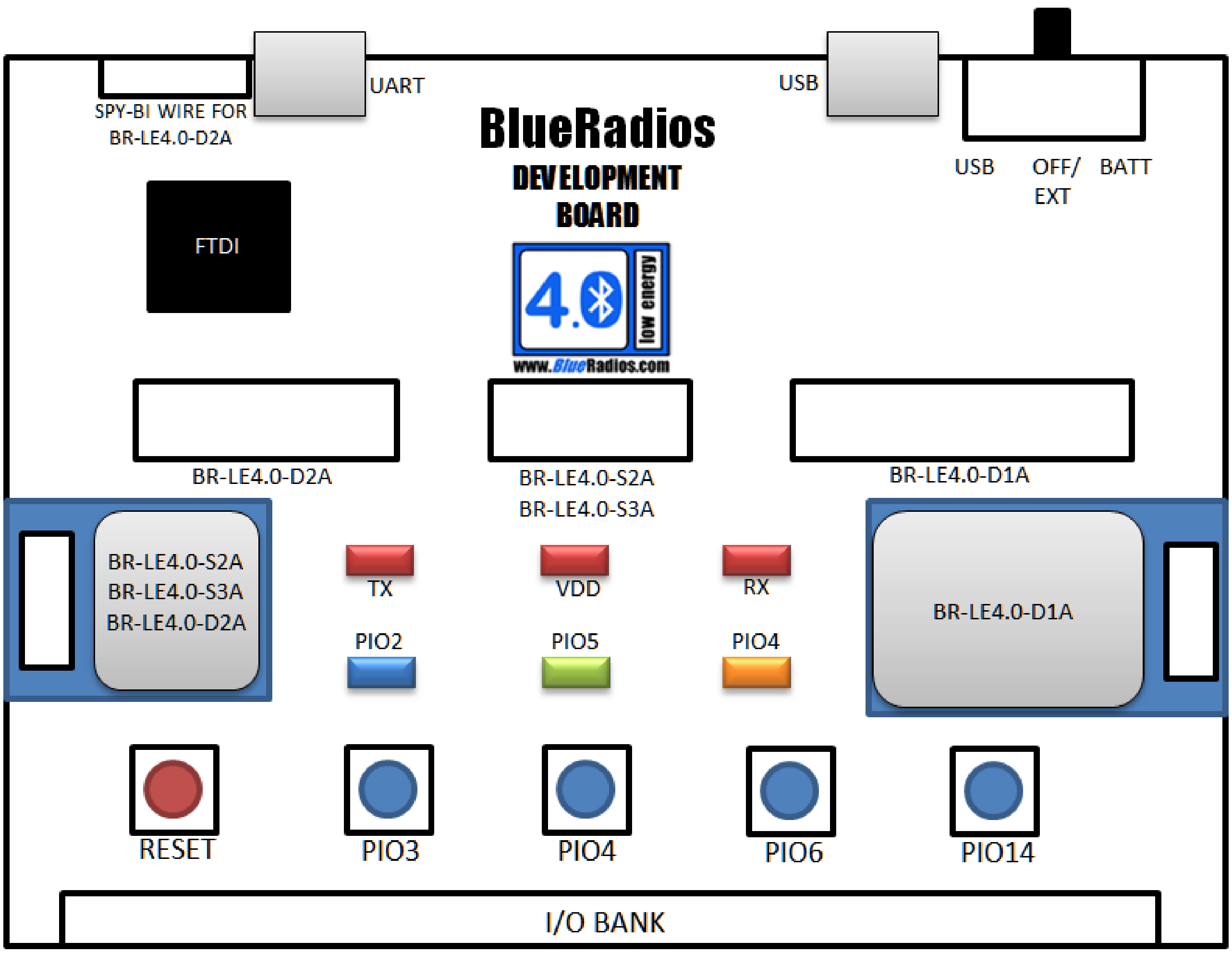
- Green LED- PIO5 Radio Status
- Blue LED – PIO2 Connection Status
- Red LED – 5V USB Power
- V- - GND (SPY-BI Wire for BR-LE4.0-D2A)
- V+ - VDD (SPY-BI Wire for BR-LE4.0-D2A)
- T – Test (SPY-BI Wire for BR-LE4.0-D2A)
- R – Reset (SPY-BI Wire for BR-LE4.0-D2A)
- FTDI – Serial to USB IC
- SW1 (optional) – Push Button, Connected to PIO4
- U5 (not populated) – Apple MIFI IC



5.4 *n*Blue™ BR-DEV Development Board

The BR-DEV Development Board (Dev Board) provides users with a ready to use out of the box development system that's easily customizable to fit the user's needs. This board is designed to support all of BlueRadios modules. LEDs are connected to PIOs 2,4,5,TX,RX, and switches are connected to PIOs 3,4,6,14 and RESET. In addition all of the PIOs are accessible on the I/O bank headers. The Dev Board has two USB connectors, one attached to a USB to Serial Converter to provide access to the UART through a PC and the other connected to the internal USB controller on the module (not supported in software yet). It also has three programming/debugging headers to support the different processors in each module.

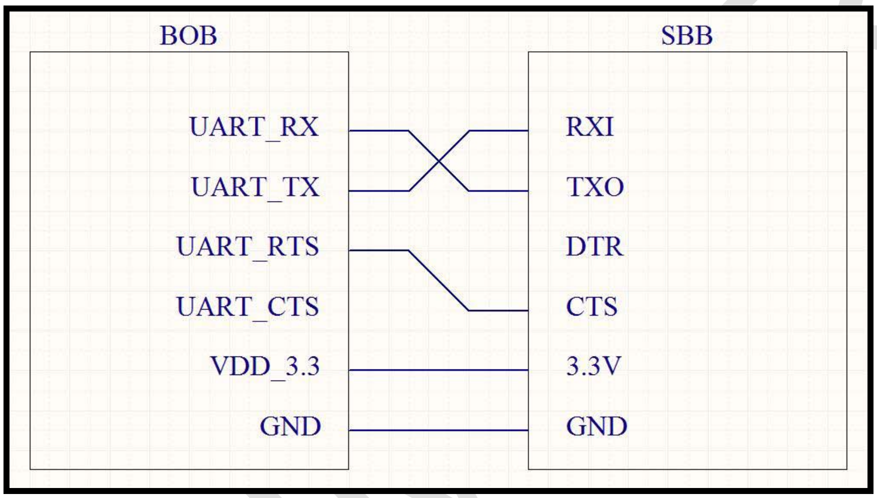
The module on the Dev Board can be powered by USB, 3.0V CR2032 battery, a Debugger or an external supply. For powering by USB set the power switch to the USB position and connect either USB port to a host. For powering by battery set the power switch to the BATT position and insert a CR2032 into the battery holder on the bottom of the board. To power through the debug header or an external supply (2.0-3.6V), set the switch to the OFF position. An external supply can be connected to the GND and VDD signals on the I/O bank headers.



5.4.1 *n*Blue™ BR-BOB Breakout Board

The BR-BOB is a breakout board with no active components other than the module, and needs to be customized to fit the user's specific needs. The user will need to supply power to the BOB, and connect to the UART to an external microcontroller or a PC. Test points are provided for every pin on the module and LED pads are available for PIO_2 (LED_1) and PIO_5 (Led_2). A debug header is also provided for connecting a BR-USB BR-LE4.0-S2 TI CC Debugger. When using it for other module types external power can be applied.

A USB Serial Breakout Board (SBB) is available to make the BOB easier to use out the box. The SBB allows the BOB to be plugged into a USB host port for testing. The SBB will supply 3.3V to the BOB, so no external power supply is needed. It can be wired to the BOB as seen in the diagram below:



6 Firmware Upgrades

6.1 Important Notes

- After loading new firmware using the ***n*Blue™** Programmer, the module will be reset back to its factory default configuration.

- Firmware upgrades should only be performed using the ***n*Blue™** Programmer, not a Debugger. After BlueRadios programs the bootloader into the modules the debug interface is locked. In order to program an ***n*Blue™** module using a Debugger it will then need to be unlocked, which will erase the entire flash including the module's bootloader, making it incapable of performing firmware updates using the ***n*Blue™** programmer. Once the bootloader has been erased it cannot be reinstalled in the field, it is a factory process only. However, custom software can be flashed using the ***n*Blue™** programmer, see the Custom Firmware section for more information.

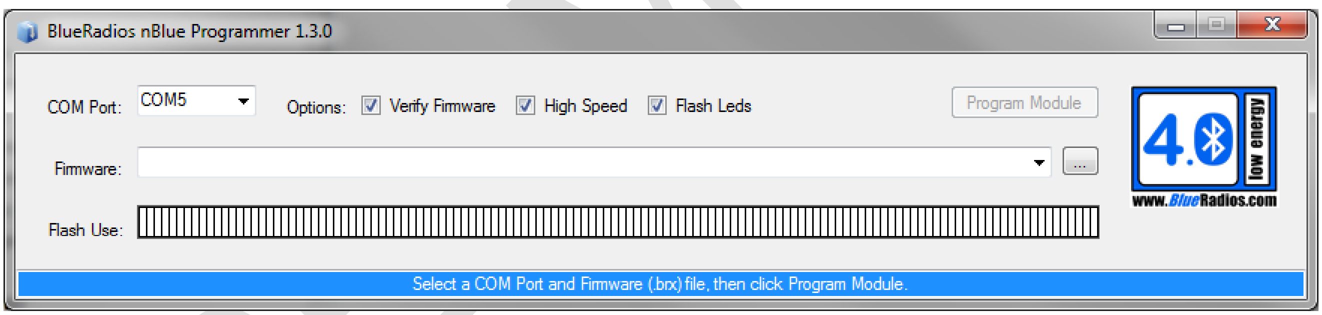
6.2 New Releases

New firmware releases will be posted at: http://www.blueradios.com/forum

6.3 *n*Blue™ Programmer

BlueRadios provides an application for Windows PCs called BlueRadios nBlue Programmer (nBlueProg.exe), which can be used to update the firmware on the evaluation boards.

nBlueProg.exe can be downloaded from: http://www.blueradios.com/forum. It requires the .NET Framework 4, which can be downloaded from: http://www.microsoft.com/net/download.aspx.



6.3.1 Upgrading the Firmware on an *n*Blue™ USB Dongle / Dev Board

1. Begin by running nBlueProg.exe.

2. Connect your Dongle/Dev Board to the PC and the COM port combo box will automatically be set to the COM port assigned to your board. If your board was connected prior to starting nBlueProg you may have to manually select the COM Port.

3. Click the "…" button to the right of the Firmware combo box to browse for the .hex file to program the device with.

4. Click the Program Module button and the module will be erased, programmed and verified.

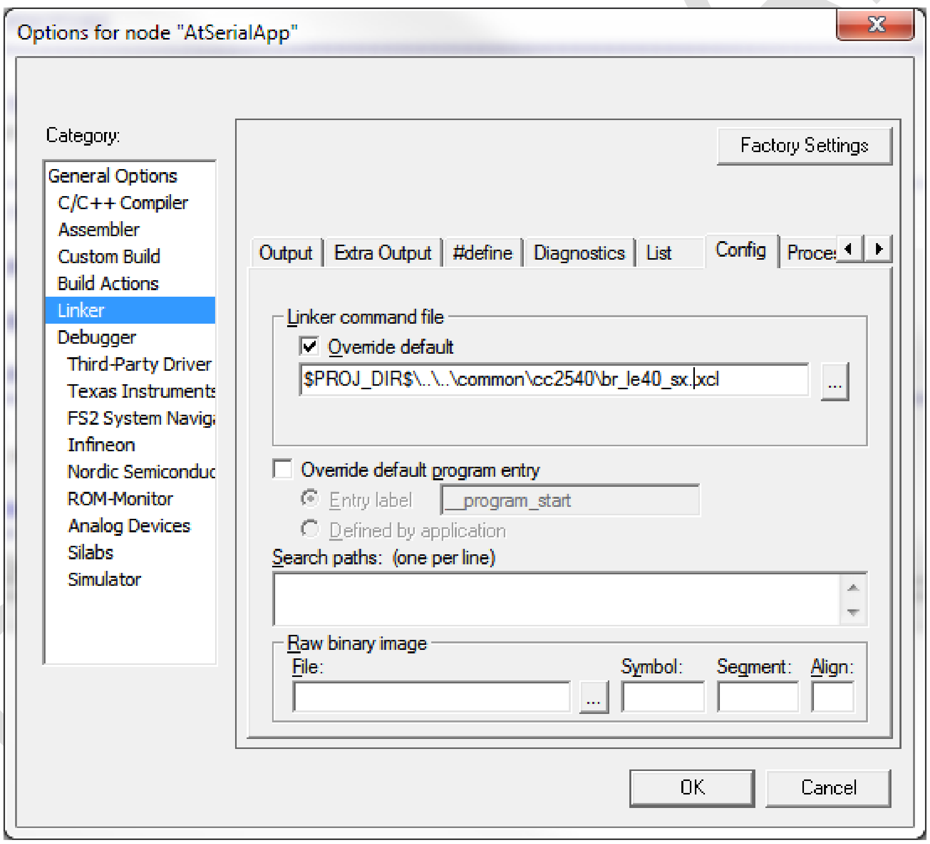
5. Upon completion, the status bar at the bottom of the window will say, "Module EC:FE:7E:##:##:## programmed succesfully! Firmware version is now #.#.#.#.#.#-#". The module will now be factory reset and running the new firmware.

6.3.2 BR-LE4.0-S2 Custom Firmware

BlueRadios provides an IAR linker command file (br_le40_sx.xcl) that allows the developer to create custom firmware using the IAR compiler that can be downloaded with the nBlue Programmer. This file can be downloaded from: http://www.blueradios.com/forum.

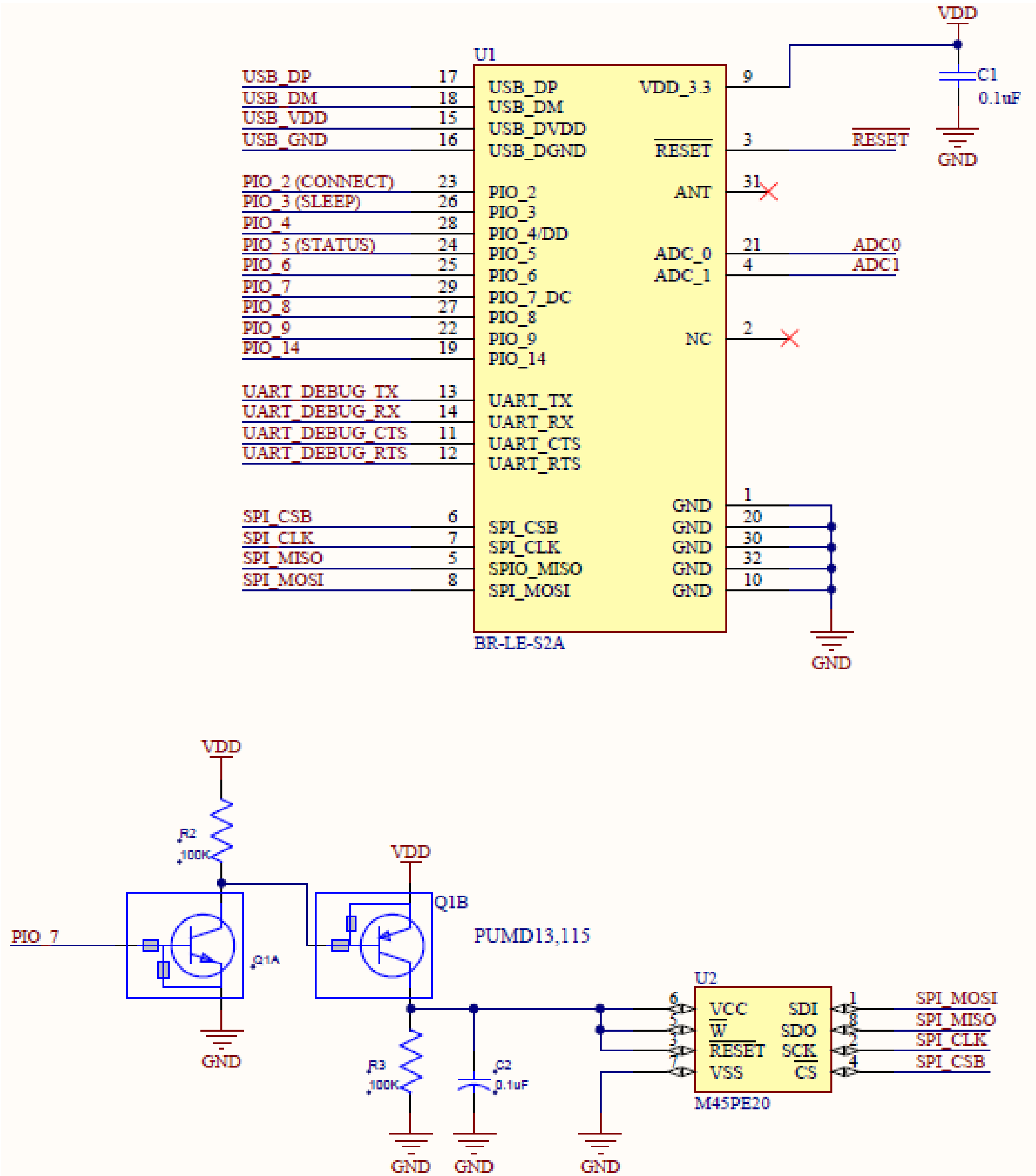
Once custom firmware has been loaded into a module the *n*Blue™ Programmer will not be able to automatically put the module into upgrade mode. The module will have to be manually put into upgrade mode by setting PIO_14 to VDD during power up or reset and holding it at VDD until both PIO_2 and PIO_5 output VDD.

The linker command file can be changed in the IAR project options dialog in Linker->Config:



6.3.3 BR-LE4.0-S2 Over the Air (OTA) Updates

The BR-LE4.0-S2 can support OTA updates through BRSP. In order to do so it must have a 2Mb M45PE20 external flash connected to its SPI bus as shown in the following reference design.



Appendix A: Acronyms/Abbreviations

AD – Advertisement
API – Application Protocol Interface
AT – Attention
ASCII – American Standard Code for Information Interchange
BLE – *Bluetooth* Low Energy
BOB – Breakout Board
BR – BlueRadios
BR/EDR – Basic Rate/Enhanced Data Rate (classic *Bluetooth*)
BT – *Bluetooth*
COD – Class of Device
COM/COMM – Communications
CR – Carriage Return
CTS – Clear To Send
DC – Direct Current
EDR – Enhanced Data Rate
GAP – Generic Access Profile
GATT – Generic Attribute Profile
GFSK – Gaussian Frequency-Shift Keying
GND – Ground
HCI – Host Controller Interface
IP – Internet Protocol
ISM – Industrial, Scientific, and Medical
LED – Light Emitting Diode
LF – Line Feed
MCU – Microcontroller Unit
MISO – Master In Slave Out
MOSI – Master Out Slave In
NC – No Connect
PC – Personal Computer
PCB – Printed Circuit Board
PIN – Personal Identification Number
RF – Radio Frequency
PIO – Programmable Input/Output
RSSI – Received Signal Strength Indication
RST – Reset
RTS – Ready To Send
RX – Receive
SBB – Serial Breakout Board
SPI – Serial Protocol Interface
SPP – Serial Port Profile
TDMA – Time Division Multiple Access
TX – Transmit
UART – Universal Asynchronous Receiver/Transmitter
µs – Microseconds
USB – Universal Serial Bus
UUID – Universal Unique Identifier
V – Volts
VDD – DC Power

APPENDIX B: Extended Inquiry Response and Ad Structure Types

1. Flags

Value	Description	Bit	Information
0x01	Flags	0	LE Limited Discoverable Mode
		1	LE General Discoverable Mode
		2	BR/EDR Not Supported (i.e. bit 37 of LMP Extended Feature bits Page 0)
		3	Simultaneous LE and BR/EDR to Same Device Capable (Controller) (i.e. bit 49 of LMP Extended Feature bits Page 0)
		4	Simultaneous LE and BR/EDR to Same Device Capable (Host) (i.e. bit 66 of LMP Extended Feature bits Page 1)
		5..7	Reserved

2. Services

Value	Description	Information
0x02	16-bit Service UUIDs	More 16-bit UUIDs available
0x03	16-bit Service UUIDs	Complete list of 16-bit UUIDs available
0x04	32-bit Service UUIDs	More 32-bit UUIDs available
0x05	32-bit Service UUIDs	Complete list of 32-bit UUIDs available
0x06	128-bit Service UUIDs	More 128-bit UUIDs available
0x07	128-bit Service UUIDs	Complete list of 128-bit UUIDs available

3. Local Name

Value	Description	Information
0x08	Local Name	Shortened local name
0x09	Local Name	Complete local name

4. TX Power Level

Value	Description	Information
0x0A	TX Power Level (1 byte)	0xXX:-127 to +127dBm Note: when the TX Power Level tag is not present, the TX power level of the packet is unknown.

5. **Simple Pairing Optional OOB Tags**

Value	Description	Information
0x0D	Class of device (3 octets)	Format defined in Assigned Numbers
0x0E	Simple Pairing Hash C (16 octets)	Format defined in [Vol. 2], Part H Section 7.2.2
0x0F	Simple Pairing Randomizer R (16 octets)	Format defined in[Vol. 2], Part H Section 7.2.2

6. **Security Manager TK Value**

Value	Description	Information
0x10	TK Value	Value as used in pairing over LE Physical channel. Format defined in [Vol. 3], Part H Section 2.3

7. **Security Manager OOB Flags**

Value	Description	Bit	Information
0x11	Flag (1 octet)	0	OOB Flags Field (0 = OOB data not present, 1 = OOB data present)
		1	LE supported (Host) (i.e. bit 65 of LMP Extended Feature bits Page 1
		2	Simultaneous LE and BR/EDR to Same Device Capable (Host) (i.e. bit 66 of LMP Extended Feature bits Page 1)
		3	Address type (0 = Public Address, 1 = Random Address)
		4.7	Reserved

8. Slave Connection Interval Range

Value	Description	Information
0x12	Slave Connection Interval Range	The first 2 octets defines the minimum value for the connection interval in the following manner: $connInterval_{min}$ = Conn_Interval_Min * 1.25 ms Conn_Interval_Min range: 0x0006 to 0x0C80 Value of 0xFFFF indicates no specific minimum. Values outside the range are reserved. (excluding 0xFFFF) The second 2 octets defines the maximum value for the connection interval in the following manner: $connInterval_{max}$ = Conn_Interval_Max * 1.25 ms Conn_Interval_Max range: 0x0006 to 0x0C80 Conn_Interval_Max shall be equal to or greater than the Conn_Interval_Min. Value of 0xFFFF indicates no specific maximum. Values outside the range are reserved (excluding 0xFFFF)

9. Service Solicitation

Value	Description	Information
0x14	Service UUIDs	List of 16 bit Service UUIDs
0x15	Service UUIDs	List of 128 bit Service UUID

10. Service Data

Value	Description	Information
0x16	Service Data (2 or more octets)	The first 2 octets contain the 16 bit Service UUID followed by additional service data

11. Manufacturer Specific Data

Value	Description	Information
0xFF	Manufacturer Specific Data (2 or more octets)	The first 2 octets contain the Company Identifier Code followed by additional manufacturer specific data