# Bluetopia® Protocol Stack Information

## Introduction

Thank you for selecting the Bluetopia, the Bluetooth Protocol Stack by Stonestreet One. This release has been prepared to support the MSP430 Platform using either the CCS or IAR MSP430 tool chain. Please Contact Stonestreet One for any technical support issues.

This release includes the core protocol stack with Low Energy support, the Serial Port Profile (SPP), the Generic Attribute Profile (GATT) and various Low Energy services. Sample applications utilizing the Serial Port Profile, the Human Interface Device Profile and the Generic Attribute Profile are provided.

## Hardware/Software Requirements

- Development platform capable of supporting required CCS or IAR compilation tools
- MSP430 Development Platform (with TI CC256x or CC256xB Chipset). The following platforms are supported:
  - ez430 – RF256x
  - MSP430F5438A Experimenter Board
  - MSP430F5529 Experimenter Board

## Hardware Jumper Requirements

For running on the MSP430F5438A Experimenter Board the following jumper assignments are required to run the sample application:

- RF3 Pin 3 to RF3 Pin 6
- RF3 Pin 5 to RF3 Pin 8
- RF3 Pin 7 to RF3 Pin 10
- RF3 Pin 11 to RF3 Pin 14

The following jumper must be connected in order to receive input and output from the MSP430F548A Experimenter board Serial to USB debugging connector:

- JP5 TX
- JP5 RX
- JP4 RST

For running on the MSP430F5529 Experimenter Board the following jumper assignments are required to run the sample application:

- JP3 removed
- J12 Pin 3 to J12 Pin 4 (P1.0 to P2.0)
- J12 Pin 5 to J12 Pin 6 (P4.4 to P4.0)
- J12 Pin 7 to J12 Pin 8 (P4.5 to P4.6)
- J12 Pin 9 to J12 Pin 10 (P4.7 to P2.4)

## C/C++ Application Development Notes

This section details specific usage information on how to effectively use Bluetopia with C/C++ Compilers.

In order to include all necessary prototypes/constants, the programmer needs to only include the **SS1BTPS.h** header file (for the core library). In order to include all necessary prototypes/constants for profiles, the programmer needs to only include the **SS1BT**xxx.**h** for the specific profile (where, 'xxx" is the profile name).

Users using C++ **MUST** use the following code to include the **SS1BTPS.H** Header File in their C++ Source:

```
extern "C" {
   #include "SS1BTPS.h"
} ;
```

Users using C (and NOT C++) do NOT need to include the above code snippet and can simply include the **SS1BTPS.h** Header File as normal, for example:

```
#include "SS1BTPS.h"
```

All API structures that are used with Bluetopia at the hardware device interface level are aligned on default packing boundaries. This shouldn't pose much of a problem because most of the structures are aligned such that they would not cause too many packing problems otherwise. Lower level structures are aligned based on compiler switches as specified by the customer.

## CCS/IAR Toolchain Library Notes

The libraries contained in this distribution were compiled using either CCS or IAR MSP430 tool chain. Users simply are required to link with the **libBluetopia.a** library for core stack functionality (stored in the folder **lib\CCS** or **lib\IAR**). All other profiles and support libraries have a corresponding library directory (also named **lib\CCS** or **lib\IAR**) which contains the library required for linking.

A CCS and IAR Project is provided for the sample applications (source is included).

## Building the Sample Applications

This release is intended to be built with either Code Composer Studio IDE or IAR's Embedded Workbench for MSP430. A CCS Project is included in the \Project\CCS folder and an IAR Project is included in the \Project\IAR folder.

## Sample Application Notes

This release includes a sample application directory with source code which demonstrates the use of the Bluetopia stack regarding the several different profiles (as described below) are included.

Input and Output can be utilized by plugging in the MSP430F5438A Experimenter board Serial to USB debugging connector to a PC. The ez430-RF256x when connected to the USB stick can also communicate with a PC. The parameters for this port are:

| | |
|---|---|
| Baud: | 9600 |
| Data Bits: | 8 |
| Parity: | None |
| Stop Bits: | 1 |
| Flow Control: | None |
| Local Echo: | Off |

### SPPDemo

This application also shows how to utilize the SPP module and how to handle the different SPP callback events. This will act as either a SPP Server or SPP Client.

### HIDDemo

This application also shows how to utilize the HID module and how to handle the different HID callback events. This will act as either a HID Host or HID Device.

### A3DPDemo_SNK

This application also shows how to utilize the A2DP function of the Bluetopia core stack and the Assisted A2DP (A3DP) of the TI CC2564. This will act as a Sink device and also demonstrates the AVRCP Controller role.

### A3DPDemo_SRC

This application also shows how to utilize the A2DP function of the Bluetopia core stack and the Assisted A2DP (A3DP) of the TI CC2564. This will act as a Source device and also demonstrates the AVRCP Target role.

### A3DPMultiRoomDemo

This application shows how to utilize the A2DP function of the Bluetopia core stack and the Assisted A2DP (A3DP) of the TI CC2564. This will act as a Sink device, demonstrates the AVRCP Controller core, and includes an updated service pack for the CC2564 that allows for a special “Multi-Room” vendor-specific command to forward an existing A2DP stream to a secondary speaker running the same service pack on another CC2564.

### SPPLEDemo

This application shows how to utilize Low Energy (LE) and the GATT profile as well as the Serial Port Profile. This sample emulates using SPP over LE using the GATT Profile. This sample acts as a LE Master and LE Slave. It can also act as a SPP Server and Client.

This application also shows how to utilize the SPP module and how to handle the different SPP callback events. This will act as either a SPP Server or SPP Client and exposes the same command set for SPP as the **SPPDemo.**

### SPPLEDemo_Lite

This application is an application that showcases the SPP Server role, GATT Server role and the LE slave role. A device can connect to any device running this application (using LE or BR/EDR) and send data (using SPP or SPPLE server) to it. The device running this application will loop back any data received.

### ANPDemo

This application shows how to utilize Low Energy (LE) and the GATT profile. This sample implements the Alert Notification Profile (ANP) using the GATT profile. This sample acts as a LE Master and LE Slave.

### FMPDemo

This application shows how to utilize Low Energy (LE) and the GATT profile. This sample implements the Find Me Profile (FMP) using the GATT profile. This sample acts as a LE Master and LE Slave.

### HOGPKeyboardDemo

This application shows how to utilize Low Energy (LE) and the GATT profile. This sample implements the HID over GATT Profile (HOGP) using the GATT profile. This sample acts as a HID Keyboard device. This sample is only supported on the MSP430F5438A Experimenter Board since it uses the LCD as a keyboard.

### HRPDemo

This application shows how to utilize Low Energy (LE) and the GATT profile. This sample implements the Heart Rate Monitoring Profile (HRP) using the GATT profile. This sample acts as a LE Master and LE Slave.

### HTPDemo

This application shows how to utilize Low Energy (LE) and the GATT profile. This sample implements the Health Thermometer Profile (HTP) using the GATT profile. This sample acts as a LE Master and LE Slave.

### KeyFobDemo

This application shows emulates the CC2540 Key Fob (using LE) and supports the SPP Server role. This application can be connected over LE and will expose button press data and accelerometer data using two TI proprietary protocols. This device can also be connected to over BR/EDR and will send the same button press data and accelerometer data to any device connected over SPP.

### PASPDemo

This application shows how to utilize Low Energy (LE) and the GATT profile. This sample implements the Phone Alert Status Profile (PASP) using the GATT profile. This sample acts as a LE Master and LE Slave.

### PXPDemo

This application shows how to utilize Low Energy (LE) and the GATT profile. This sample implements the Proximity Profile (PXP) using the GATT profile. This sample acts as a LE Master and LE Slave.